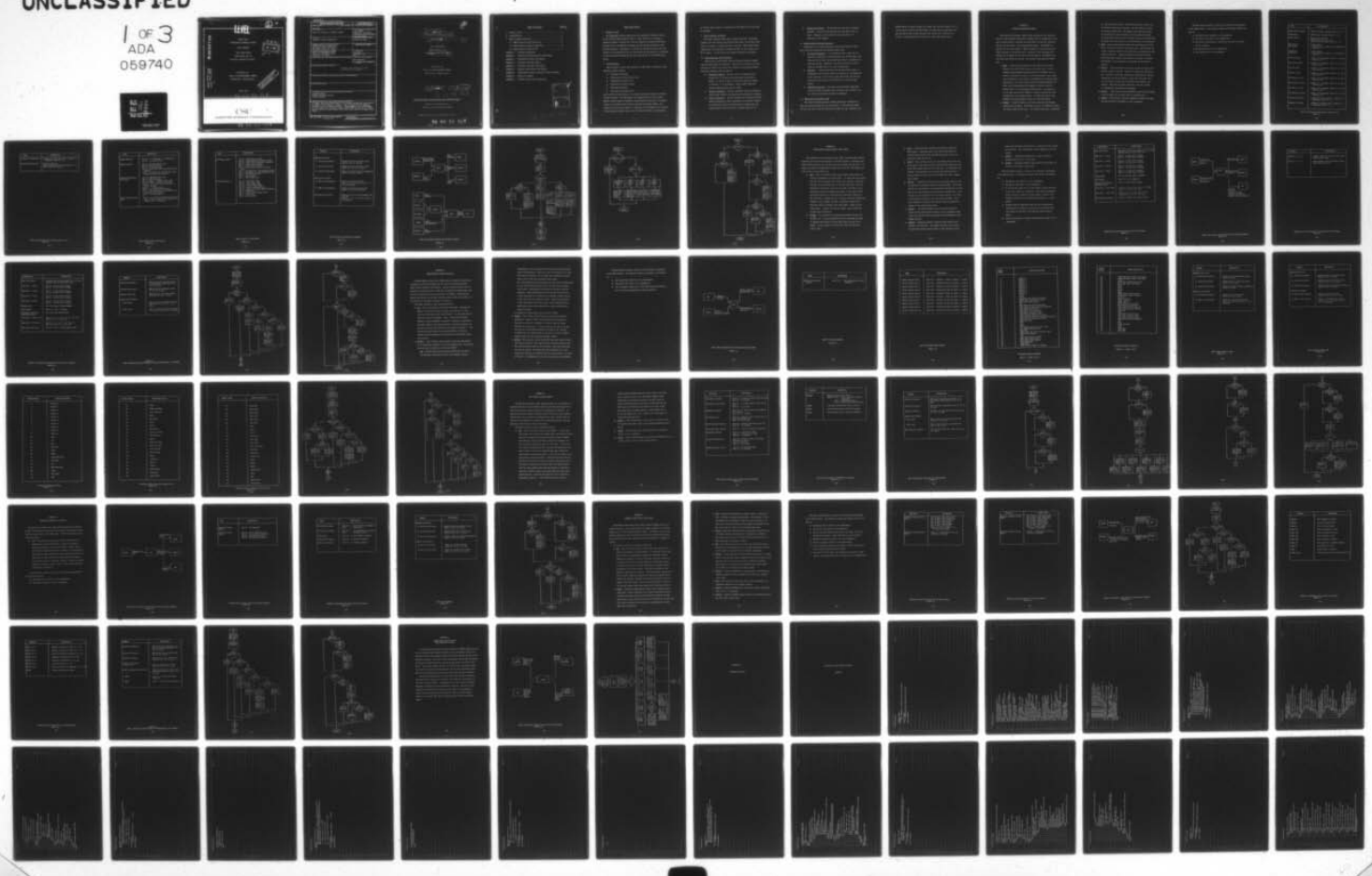
AD-A059 740 COMPUTER SCIENCES CORP HUNTINGDON VALLEY PA F/G 15/1
LAMPS SEAS SIMULATION SOFTWARE SUPPORT.(U)
JUN 78 N62269-75-C-0001
UNCLASSIFIED NL

1 OF 3
ADA
059740



LEVEL II

(2)

AD A059740

DDC FILE COPY

LAMPS SEAS

SIMULATION SOFTWARE SUPPORT

FINAL REPORT

CDRL ITEM #A004

Task Order No. 55

Contract N62269-75-C-0001

Prepared for

NAVAL AIR DEVELOPMENT CENTER

Warminster, Pennsylvania

June 1978

CSC

COMPUTER SCIENCES CORPORATION

UNCLASSIFIED
SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER Not Cited	2. GOVT ACCESSION NO.	3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle) LAMPS SEAS Simulation Software Support		5. TYPE OF REPORT & PERIOD COVERED Final Report Sept 1977 - June 1978
		6. PERFORMING ORG. REPORT NUMBER CDRL Item #A004
7. AUTHOR(s) Computer Sciences Corporation		8. CONTRACT OR GRANT NUMBER(s) N62269-75-C-0001
9. PERFORMING ORGANIZATION NAME AND ADDRESS Computer Sciences Corporation 101 Mason's Mill Business Park Huntingdon Valley, PA 19006		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS
11. CONTROLLING OFFICE NAME AND ADDRESS Naval Air Development Center Street and Jacksonville Rds. Warminster, PA 18974		12. REPORT DATE June 1978 13. NUMBER OF PAGES 231
14. MONITORING AGENCY NAME & ADDRESS(if different from Controlling Office)		15. SECURITY CLASS. (of this report) Not Classified 15a. DECLASSIFICATION/DOWNGRADING SCHEDULE
16. DISTRIBUTION STATEMENT (of this Report) APPROVED FOR PUBLIC RELEASE- DISTRIBUTION UNLIMITED		
17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)		
18. SUPPLEMENTARY NOTES		
19. KEY WORDS (Continue on reverse side if necessary and identify by block number) Software modules Simulated avionics devices Profile listings		
20. ABSTRACT (Continue on reverse side if necessary and identify by block number) The report contains a description of the software modules that were written in support of the LAMPS SEAS Program. Most of these modules represent simulated avionics hardware devices. In addition, there is an Input/Output Executive and a Data Collection module. Included in the body of the report are descriptions of each module, flow charts, and a profile listing of each module.		

410 506

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE
S/N 0102-014-6601

UNCLASSIFIED
SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

(2)

(6) LAMPS SEAS

SIMULATION SOFTWARE SUPPORT.

(9) FINAL REPORT. Sep 77-Jun 78.

CDRL ITEM #A004

Task Order No. 55

(15) Contract N62269-75-C-0001

Prepared for

NAVAL AIR DEVELOPMENT CENTER

Warminster, Pennsylvania

(11) June 1978 (12) 236p.

COMPUTER SCIENCES CORPORATION

101 Masons Mill Business Park

Huntingdon Valley, Pennsylvania 19006

Major Offices and Facilities Throughout the World

TABLE OF CONTENTS PAGE NO.

LAMPS FINAL REPORT

1 SUMMARY OF TASK

The LAMPS SEAS software support task was performed to simulate several of the avionics remote terminals (RT's). Their function is to supply a simulated hardware interface for those RTs between the Avionics Operational Program (AOP) in the AN/AYK-14 computer and the overall simulation in the CDC 6600 computer. Additionally, a routine was included to provide for the routing of AOP commands and data to the various RTs and provide for collection of data that passes through the data bus from the AOP to the CDC-6600 simulation.

2 ACCOMPLISHMENTS

The end product of this task provides for LAMPS SEAS a simulation of the following hardware modules:

- Converter Multiplexer
- Communication System Control Group
- Navigational Interface Unit
- Multifunction Control System
- MAD Signal Processor
- Ordnance Launch Control Set.

Also included in this product is a combined input/output Executive and Data Collection module. The development and implementation of each of these modules includes general flowcharts, design/coding, definition of variables unique to each module, batch drivers and batch environment debug. Descriptions of the various modules is contained in the Appendices (by direction of the cognizant engineer work on the MAD Signal Processor was terminated

as of April 1978, however, a description of the module as of that date is included).

3 REMOTE TERMINAL PROCESSING

The Remote Terminals (RTs) share a common structure - processing commands from the Avionics Operational Program (AOP), RT internal operations, and output of status and data to the AOP. Within these three subdivisions the processing of information which is not unique to the RT is similar. In this section the common processing is described.

3.1 Remote Terminal Input Processing

Each RT has an input buffer from the AOP which contains command words and data words for processing. Those RTs in the 40 ms loop have non-circular buffers, the others have circular or wrap-around buffers. The common transfers from the AOP are:

- Initialize Terminal - The RT is put in a quiescent state during which all internal and input/output processing is halted. If the RT is under a "self-test" when the command is received, it is terminated. Output buffer flags that may have been previously set are reset.
- Initiate Processing - The RT-s quiescent state is terminated, allows normal internal and input/output processing to commence.
- Initiate Self-Test - The RT is placed in a simulated self-test state by ceasing all normal processing, the self-test counter is initialized, and the RT status word is inserted into the output buffer to the AOP with the "receive busy" bit set.

- Normal Data Transfer - The RT determines the type of transfer required. If an RT to AOP transfer, the "data sent" flag is reset. Otherwise, the RT processes the input data as is required by that terminal.

3.2 Remote Terminal Internal Operations

In addition to its unique operations, each RT processes the self-test, BIT status monitoring and quiescent state check.

- Bit Status Monitoring - If a transition from a logic zero to logic one is detected in the BIT status word, the RT status word (with the T/F bit set) and the BIT status word is transferred to the output buffer. Otherwise, the new BIT status is saved to reset any possible logic 1 to logic 0 transitions.
- Self-Test - During a self-test the RT's self-test counter is decremented and the result tested for completion. On completion of the self-test, the RT status (with the T/F bit set) and the current BIT status words are transferred to the RT's output buffer.
- Quiescent State Test - Upon entering the internal operations section, the RT's quiescent state is tested, and if found to be on, further operational processing is bypassed.

3.3 Remote Terminal Output to AOP

The output processing section involves testing for completion of previous output acceptance by the IIU and AOP. If the previous output has not been taken, then appropriate error flags are set, otherwise, the

PACKPP routine is called to pack the current data into the format of 2-1/2 -16 bit words per 60 but CDC 6600 words, a header word is constructed for the PP program, and the data available and "status sent" flags are set. If data is also being sent this cycle, the "data sent" flag is set.

APPENDIX A

CONVERTER MULTIPLEXER MODULE

The Converter Multiplexer (CMUX) module provides for the simulated control of AOP commands, formating of display data for refreshing the ATO and so display buffers, building the data buffer for the AOP, reading of ATO and SO hook signals, and computes TACAN signals. The modules is divided into two main routines - CMUX1 (resident in the 40 ms loop) and CMUX2 (resident in the 200 ms loop). Basically, CMUX1 processes AOP commands and refreshes the ATO and SO display buffers, and CMUX2 builds the output data buffer to the AOP. The routines comprising the module are:

- CMUX1 - This routine processes all AOP commands to the CMUX module and is the driving routine in the 40 ms loop. The AOP commands, MODE/DISCRETE DATA and NORMAL DATA TRANSFER, are processed as described in section 3, with the exception that normal data transferred to the module are transferred to a holding buffer. Additionally, the module accepts the MULTI-MESSAGE TRANSFER command, and on encountering this command, transfers the multi-message data to a multi-message holding buffer. On completion of its input buffer scan, CMUXCDT is called to process all data transferred to the holding buffers. On regaining control from CMUXCDT. Helo heading data is saved for later processing.
- CMUXCDT - Called by CMUX1, this routine processes the holding buffers built by CMUX1. CMUX header word one is decoded and tested for buffer count and IPL data. If a buffer count error is detected

or a bad IPL address exists, the Header Work Error flag is set, and further processing is bypassed. The data block is scanned for acoustic display data, MAD display data, ATO display data, or SO display data and transferred to the appropriate buffer. If during this process an error in header word two is encountered, the Header Work Error flag is set and processing is discontinued.

- CMUX2 - This routine performs CMUX output processing to the AOP and is the driver routine in the 200 ms loop. Upon entering the routine, self-test and BIT status processing is performed as described in Section 3. Routine CMXDATA is called to build the CMUX normal data transfer output buffer to the AOP. On return from CMXDATA, output processing is performed as described in Section 3.
- CMXDATA - Called by CMUX2, this routine builds the normal data transfer buffer to the AOP. The data inserted into the buffer are: electronic altimeter, TACAN range, TACAN bearing, ATO hook coordinates, SO hook coordinates, pitch sine and cosine, roll sine and cosine, heading sine and cosine (four samples, two sources), indicated air speed, barometric altitude, outside air temperature, and latest MAD conversion.
- MUXPACK - This routine called by CMXDATA, packs (sign extended) a real variable into a binary value of requested size.
- KZSCOMP - This routine, called by CMXDATA, transforms the input variable from one's complement to two's complement.

The CMUX module performs a subset of the functional requirements of the CMUX hardware. The disparity between the hardware and the software is:

- The Master Clear function is not implemented.
- The manual "self-test" is not implemented.
- The Initial Program Load is tested but the actual load data is not processed.
- The AOP backup mode is not implemented.
- The backup CMUX is not implemented.

WORD	DESCRIPTION
Electronic Altimeter	Binary altitude, unsigned (bits 12 - 0)
TACAN Range	TACAN range, unsigned (bits 15 - 0)
TACAN Bearing	Bearing sign (bit 15) Status code (bits 14 - 13) TACAN bearing, unsigned (bits 12 - 1)
ATO STICK-X ATO STICK-Y	X Displacement Y Displacement + (signed, 2's complement) (bits 15 - 6)
SO STICK-X SO STICK-Y	X Displacement Y Displacement + (signed, 2's complement) (bits 15 - 6)
NIU Pitch Sine	Signed, 2's complement (bits 15 - 2, bit 0 validity bit)
NIU Pitch Cosinge	Signed, 2's complement (bits 15 - 2, bit 0 validity bit)
NIU Roll Sine	Signed, 2's complement (bits 15 - 2, bit 0 validity bit)
NIU Roll Cosine	Signed, 2's complement (bits 15 - 2, bit 0 validity bit)
NIU Heading 1 Sine	Signed, 2's complement (bits 15 - 2, bit 0 validity bit)
NIU Heading 1 Cosine	Signed, 2's complement (bits 15 - 2, bit 0 validity bit)
NIU Heading 2 Sine	Signed, 2's complement (bits 15 - 2, bit 0 validity bit)
NIU Heading 2 Cosine	Signed, 2's complement (bits 15 - 2, bit 0 validity bit)
Indicated Airspeed	Indicated airspeed (knots) - signed, 2's complement (bits 15 - 6)
Barometic Altitude	Barometric Altitude (feet), signed, 2's complement (bits 15 - 6)

CMUX to AOP Message Word Formats (Page 1 of 2)
Table A-1

WORD	DESCRIPTION
Outside Air Temperature	Outside Air Temperature (°C), signed, 2's complement (bits 15 - 6)
Latest MAD Conversion	Sign bit (bit 15) Digitized MAD Analog Signal (bits 14 - 6) New/Old MAD data (bit 0)

CMUX to AOP Message Word Formats (Page 2 of 2)
Table A-1

WORD	DESCRIPTION
Header Word One	Bit 15, 1 = IPL data, 0 = display data Bits 8 - 0, Buffer count
Header Word Two	Bit 15, Acoustic Display Data Bit 14, ATO Display Data Bit 13, SO Display Data Bit 12, Header Word Three has MAD Display Command Bits 7 - 0, Number of words between this header Word Two and the next.
Acoustic/MAD Header Word Three	Bit 15, LOFAR/DIFAR ALI data Bit 14, DEMON ALI data Bit 13, Ranger - Doppler Raster data Bit 12, Ranger - Bearing Raster data Bits 11 - 7, Active Block Count for a Raster display Bits 6 - 5, Display zone Bits 4 - 3, Display zone blanking Bit 2 - Following data is continuation of previously received data Bits 1 - 0, MAD scaling
Buffer Address Header Word	Bits 15 - 0, Follows header word one during IPL, or header word two when ATO or SO display data is indicated.

AOP to CMUX Header Word Formats

Table A-2

WORD	DESCRIPTION
BIT Status Word 1	Bit 15 - A/D Converter Fault Bit 14 - Synchro/Digital Converter Fault Bit 13 - Power Off-On Transient Occurred Bit 12 - Block Count Anomaly Bit 6 - TACAN Interface Fault Bit 5 - Electronic Altimeter Interface Fault Bit 4 - SO Symbol Gen. and Interface Fault Bit 3 - ATO Symbol Gen. and Interface Fault Bit 2 - Hook Control Fault Bit 1 - Syncro Fault Bit 0 - Vital CMUX Fault
BIT Status Word 2	Bit 15 - CMUX Temp. High Bit 14 - SO CD Temp. High Bit 13 - ATO CD Temp. High Bit 12 - Invalid Header Word Bit 11 - Memory Checksum Invalid Bit 10 - Converter Display Buffer Overflow Bit 3 - TACAN Fault Bit 2 - Electronic Altimeter Fault Bit 1 - SO CD Fault Bit 0 - ATO CD Fault

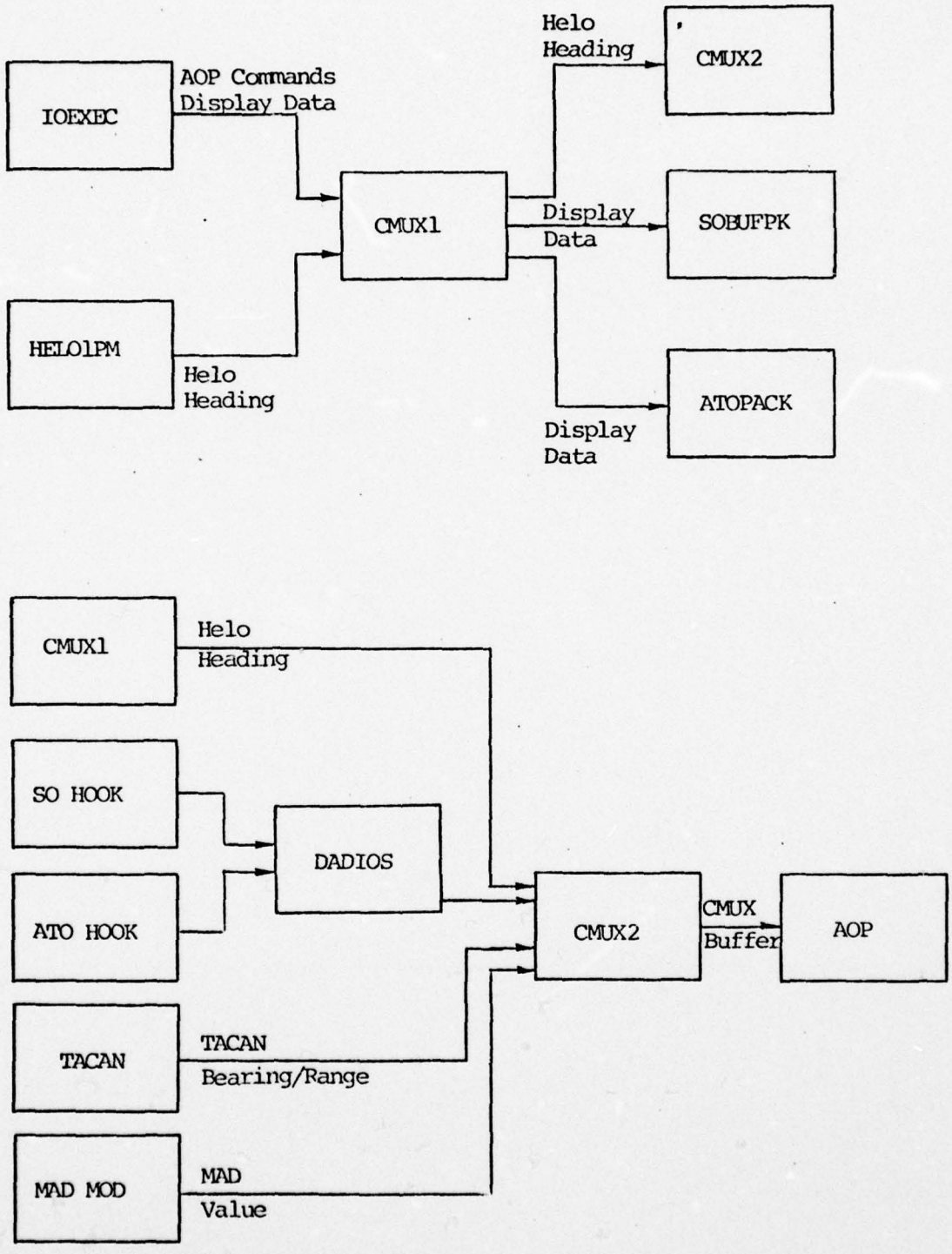
CMUX to AOP Bit Status Words

Table A-3

COMMAND	DESCRIPTION
Mode/Discrete Data	
• Initialize Terminal	Module placed in quiescent state (bits 0, 11, 12, 14 set)
• Initiate Self-Test	Module placed in self-test state (bits 0, 1, 11, 17, 14 set)
• Initiate Processing	Modules internal processing commenced (bits 2, 11, 12, 14 set)
Normal Data Transfer	
• AOP to CMUX Transfer	Module to receive AOP data (bits 5, 11, 12, 14 set; bits 0 - 4 set as required)
• CMUX to AOP Transfer	Module to transfer data to AOP (bits 5, 10, 11, 12, 14 set)
Multi-Message Transfer	Module to receive multi-message data from AOP (bits 6, 7, 11, 12, 14 set; bits 0 - 4 as required)

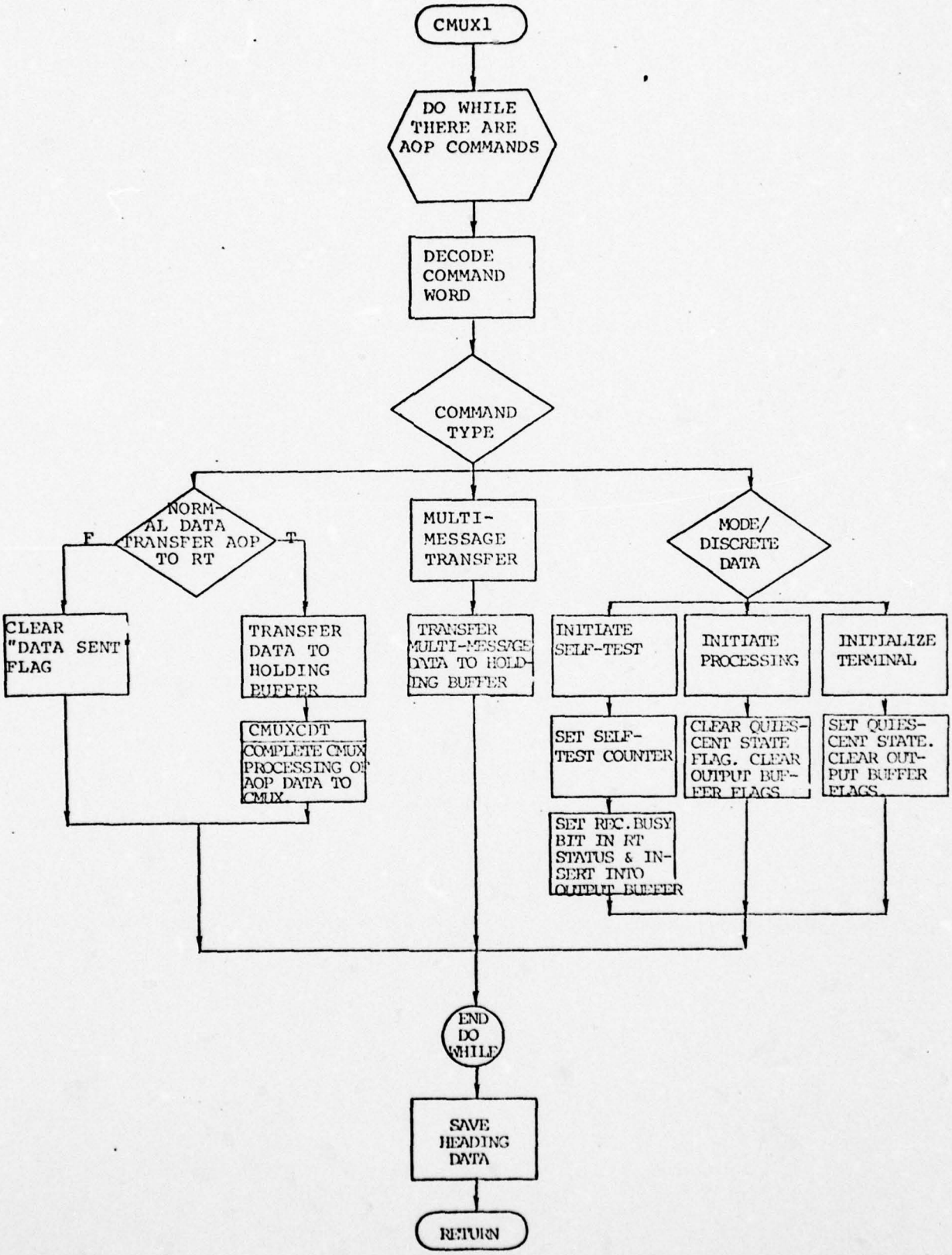
AOP to Converter Multiplexer Commands

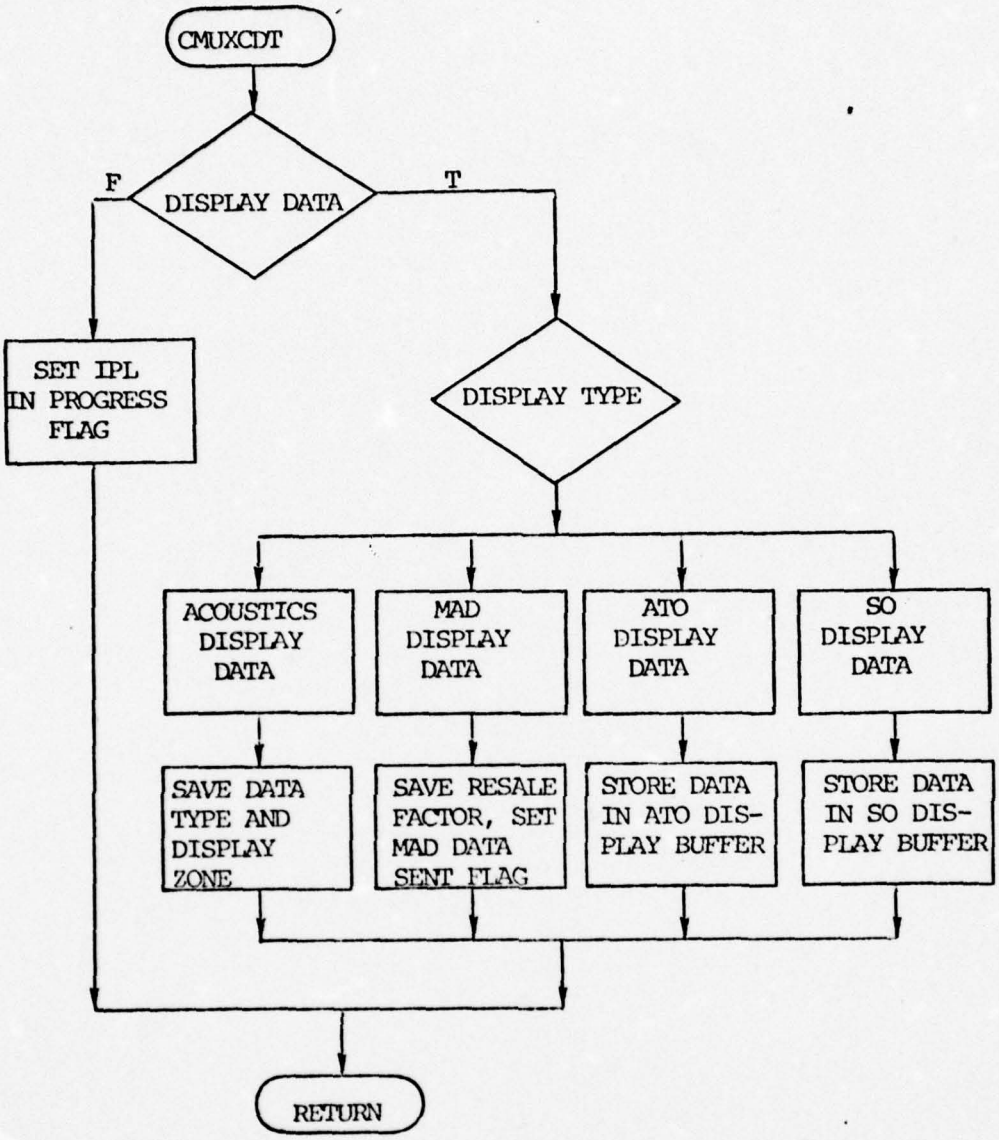
Table A-4

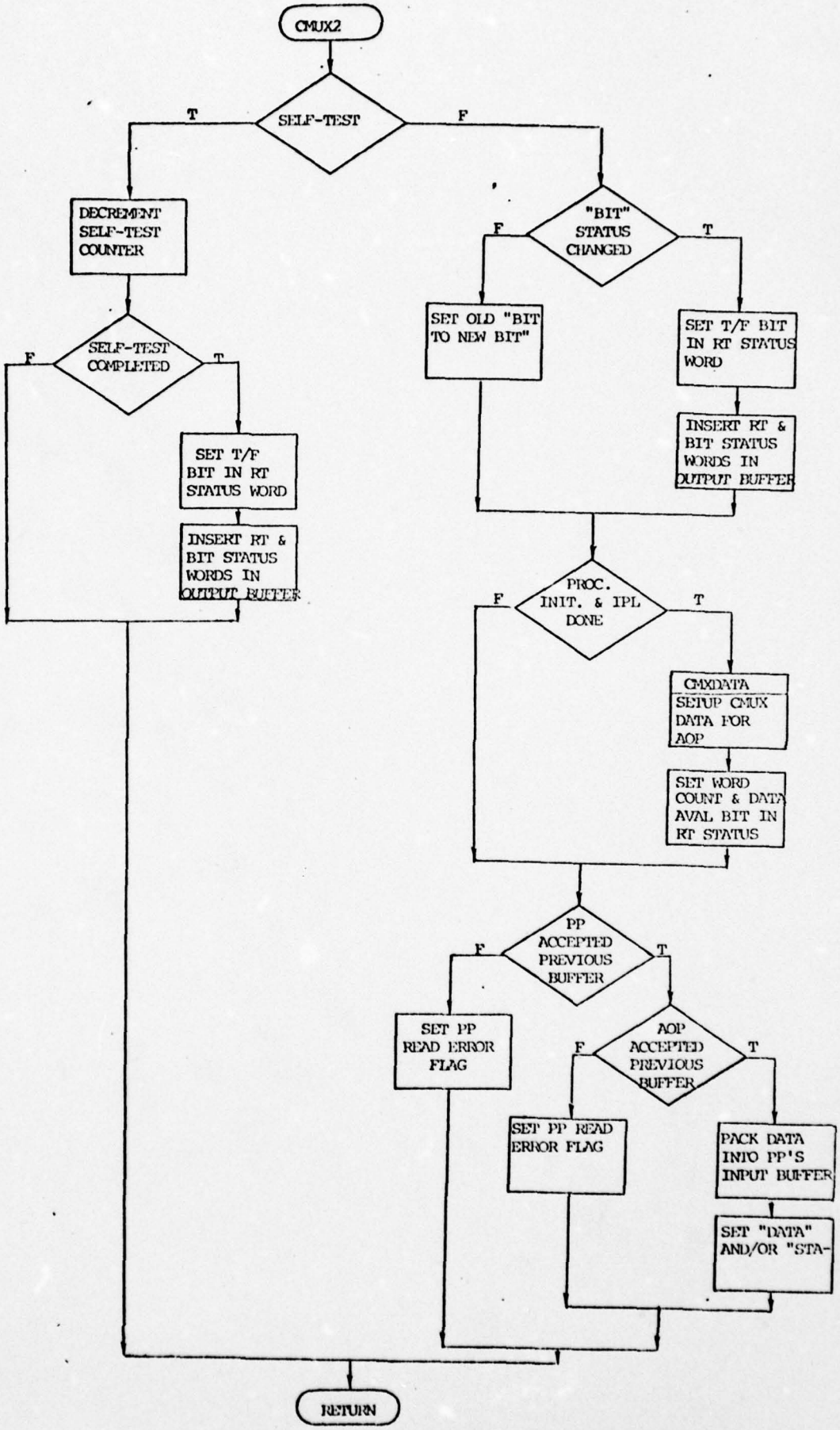


CMUX Input/Output to/from AOP and Other Modules

Figure A-1


CMUX1
DO WHILE THERE ARE AOP COMMANDS
DECODE COMMAND WORD
COMMAND TYPE
NORM-AL DATA TRANSFER AOP TO RT
F
T
MULTI-MESSAGE TRANSFER
MODE/ DISCRETE DATA
CLEAR "DATA SENT FLAG
TRANSFER DATA TO HOLDING BUFFER
CMUXCDT
COMPLETE CMUX PROCESSING OF AOP DATA TO CMUX.
TRANSFER MULTI-MESSAGE DATA TO HOLDING BUFFER
INITIATE SELF-TEST
INITIATE PROCESSING
INITIALIZE TERMINAL
SET SELF-TEST COUNTER
CLEAR QUIES-CENT STATE FLAG. CLEAR OUTPUT BUF-FER FLAGS
SET QUIES-CENT STATE. CLEAR OUT-PUT BUFFER FLAGS
SET REC. BUSY BIT IN RT STATUS & IN-SERT INTO OUTPUT BUFFER
END DO WHILE
SAVE HEADING DATA
RETURN


CMUXCDT
DISPLAY DATA
F
T
SET IPL IN PROGRESS FLAG
DISPLAY TYPE
ACOUSTICS DISPLAY DATA
MAD DISPLAY DATA
ATO DISPLAY DATA
SO DISPLAY DATA
SAVE DATA TYPE AND DISPLAY ZONE
SAVE RESALE FACTOR, SET MAD DATA SENT FLAG
STORE DATA IN ATO DISPLAY BUFFER
STORE DATA IN SO DISPLAY BUFFER
RETURN


CMUX2
SELF-TEST
T
F
DECREMENT SELF-TEST COUNTER
SELF-TEST COMPLETED
F
T
SET T/F BIT IN RT STATUS WORD
INSERT RT & BIT STATUS WORDS IN OUTPUT BUFFER
"BIT" STATUS CHANGED
F
T
SET OLD "BIT TO NEW BIT"
SET T/F BIT IN RT STATUS WORD
INSERT RT & BIT STATUS WORDS IN OUTPUT BUFFER
PROC. INIT. & IPL DONE
F
T
CMXDATA
SETUP CMUX DATA FOR AOP
SET WORD COUNT & DATA AVAL BIT IN RT STATUS
PP ACCEPTED PREVIOUS BUFFER
F
T
SET PP READ ERROR FLAG
AOP ACCEPTED PREVIOUS BUFFER
F
T
SET PP READ ERROR FLAG
PACK DATA INTO PP'S INPUT BUFFER
SET "DATA" AND/OR "STA-
RETURN

APPENDIX B

COMMUNICATION SYSTEM CONTROL GROUP MODULE

The Communication System Control Group (CSCG) software module handles simulation of the essential elements of the CSCG hardware. The module provides simulated control over the UHF radio and sonobuoy receiver units. It formats and outputs simulated CSCG status data to the AOP. The module consists of nine routines which are:

- CSCG - This is the main routine which directs and controls all functions performed in the CSCG module. It scans the input buffer from the AOP for command words (Normal Data Transfer, Initialize Terminal, Initiate Processing, or Initiate Self-Test), sets the appropriate operational status of the module, and sets an error flag should an invalid command word appear in the input stream. Self-test processing is then handled as described in Section 3. The CSCG module, through the use of several additional subroutines (CONTROL, UDICP, PERIPHL, SONOINF and DTOAINF) handles those functions which are unique to the CSCG module. These are described below. Finally, the routine handles output processing as described in Section 3.
- CSCGNDP - In the event of a Normal Data Transfer from the AOP to CSCG, this subroutine handles the task of removing the block of command (data) words from the input buffer for use by the module. If the transfer is CSCG to AOP, then the data sent flag is reset.

- UDICP - Subroutine UDICP updates inputs from the simulated CSCG Control - Indicator panel in the form of UHF mode and channel selection and resets the appropriate bits in the outgoing data words for the AOP.
- HEADER - This routine sets bits in the header word of the outgoing status (data) word block to indicate which outgoing status words have changed since the last CSCG call. It also checks for changes in the remainder of the status word block and sets a flag to indicate to the main routine (CSCG) when such a change has occurred.
- PERIPHL - Subroutine PERIPHL updates the present status of the D/L mode and the sonobuoy receivers signal strength. The signal strength is calculated by first determining the distance from the helo to the buoy assigned to the particular receiver unit. This distance is then associated with an integer (from 0 to 7) using a step-function approach to give the signal strength. After such information is determined, PERIPHL makes the appropriate changes in the outgoing data words.
- SONOINF - Subroutine SONOINF obtains the switch function values for the eight sonobuoy receiver units, translates these values into channel numbers and passes this information to the SONOBOY routine.
- DTOAINF - Subroutine DTOAINF calculates OTPI bearing, when the UFH is in OTPI mode. The routine searches for an active, in-water buoy with RF channel equal to that indicated in the

keyset and calculates the bearing to it relative to the current helo position. This information is then supplied to the DTOA module.

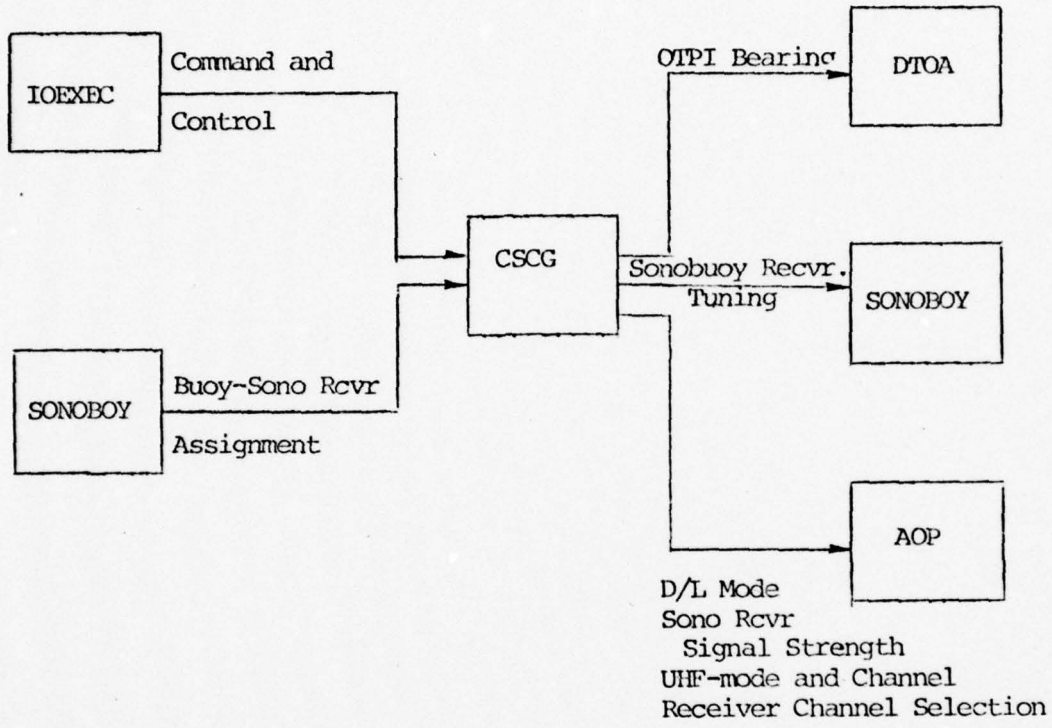
- SETABIT - Subroutine SETABIT sets a given bit within a specified word to 0 or 1 as desired.
- READBIT - Subroutine READBIT right-justifies and returns the value of a bit within a given word.

The CSCG module performs a subset of the functional requirements of the CSCG hardware. The disparity between the software and hardware is:

- The Master Clear function is not implemented.
- The manual "self-test" is not implemented.
- Of the two UHF radios only the UHF channel and mode settings of the UHF-1 radio are implemented. UHF antenna assignment, keyset loading of channel-frequency memory, squelch, tone, volume and CASS down-link command processing are not implemented.
- Sonobuoy Receiver processing does not route audio output signals from the sonobuoy receiver units to the data link, nor provide for the use of the sonobuoy receiver backup panel.
- The IFF interrogator, internal communications system are not implemented.

STATUS WORD	DESCRIPTION
Cmd. Word Status	Indicates which status words (1 to 16) have changed since the last CSCG call.
Sono Rcvr - 1 Chan	Bits 0 - 4 sono rcvr A tuning. Bits 8 - 12 sono rcvr B tuning.
Sono Rcvr - 1 Chan	Bits 0 - 4 sono rcvr E tuning. Bits 8 - 12 sono rcvr F tuning.
Sono Rcvr - 2 Chan	Bits 0 - 4 sono rcvr C tuning. Bits 8 - 12 sono rcvr D tuning.
Sono Rcvr - 2 Chan	Bits 0 - 4 sono rcvr G tuning. Bits 8 - 12 sono rcvr H tuning.
Switch Cmds.	Bit 15 UHF mode (AUTO or MANUAL)
UHF Channel	Bits 0 - 7 UHF-1 channel.
Peripheral Equipment Operating Stat.	Bit 0 D/L Mode (ASW or ASST)
Sono Rcvr - 1 Sig. Stg.	Bits 0 - 2, 4 - 6, 8 - 10, 12 - 14 Sono Rcvrs. F, E, B, A sig. stg.
Sono Rcvr - 2 Sig. Str.	Bits 0 - 2, 4 - 6, 8 - 10, 12 - 14 Sono Rcvrs. H, G, D, C sig. stg.
UHF Radio Mode Status	Bits 4, 15 UHF-1 mode (OTPI or ADF)

Communication System Control Group to AOP Formats
Table B-1



Inputs and Outputs to and from the AOP and Other Modules
Figure B-1

VARIABLE	DESCRIPTION
IRFCH(K) K = 1, 8	Channel number for each of eight sonobuoy receiver units.
ICH (K) K = 1, 8	Chute number of buoy assigned to sonobuoy receiver unit K.

Communication System Control Group/Sonobuoy Data Formats
Table B-2

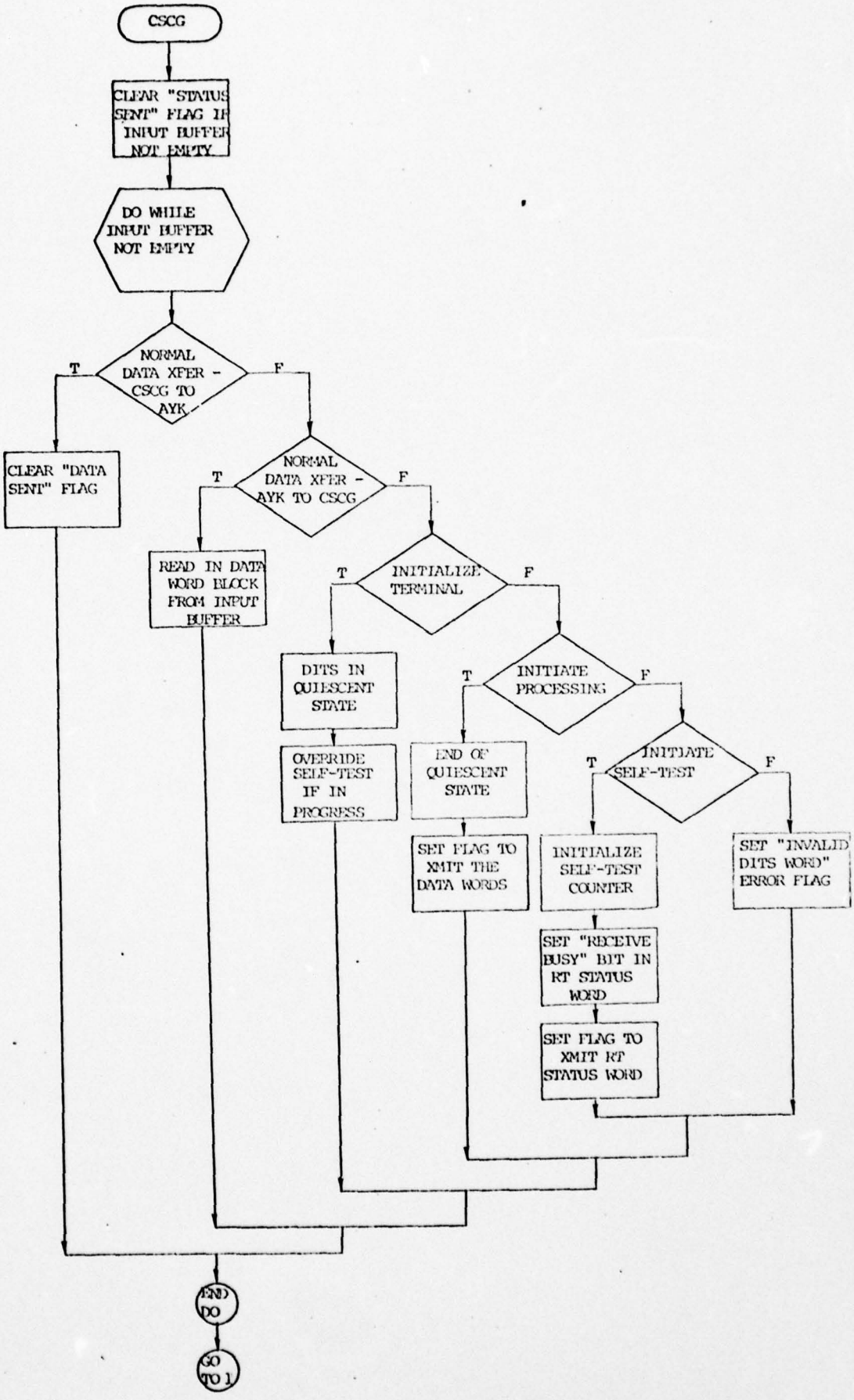
COMMAND WORD	DESCRIPTION
Cmd. Word Status	Indicates which status words (1 to 16) have changed since the last CSCG call.
Sono Rcvr - 1 Chan	Bits 0 - 4 sono rcvr A runing. Bits 8 - 12 sono rcvr B tuning.
Sono Rcvr - 1 Chan	Bits 0 - 4 sono rcvr E tuning. Bits 8 - 12 sono rcvr F tuning.
Sono Rcvr - 2 Chan	Bits 0 - 4 sono rcvr C tuning. Bits 8 - 12 sono rcvr D tuning.
Sono Rcvr - 2 Chan	Bits 0 - 4 sono rcvr G tuning. Bits 8 - 12 sono rcvr H tuning.
Switch Cmds	Bit 15 UHF mode (AUTO or MANUAL)
UHF Channel	Bits 0 - 7 UHF - 1 channel
Peripheral Equipment Operating Status	Bit 0 D/L Mode (ASW or ASST)
Sono Rcvr - 1 Sig. Stg.	Bits 0 - 2, 4 - 6, 8 - 10, 12 - 14 sono rcvrs F, E, B, A sig. stg.
Sono Rcvr - 2 Sig. Str.	Bits 0 - 2, 4 - 6, 8 - 10, 12 - 14 Sono rcvrs H, G, D, C sig. stg.
UHF Radio Mode Status	Bits 4, 15 UHF - 1 mode (OTPI or ADF)

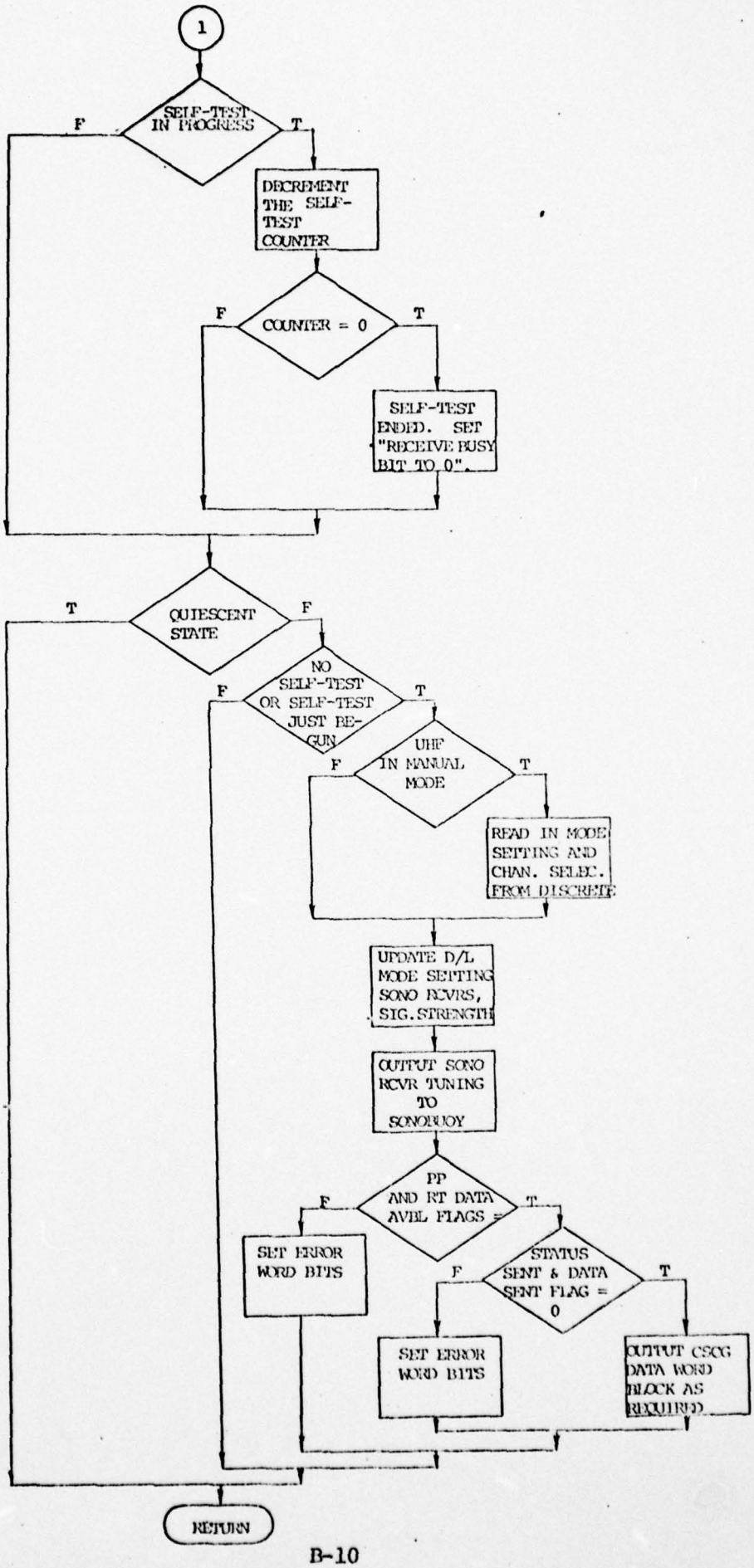
IOEXEC to Communication System Control Group Data Formats

Table B-3

COMMAND	DESCRIPTION
Initialize Terminal	CSCG forced into quiescent state, all processing halted (bits 13, 12, 11, 0 set).
Initiate Processing	CSCG processing commenced (bits 13, 12, 11, 2 set).
Initiate Self-Test	CSCG Built in Test (BIT) forced (bits 13, 12, 11, 1, 0 set).
Normal Data Transfer	
AOP to CSCG	CSCG to process new data from the AOP (bits 13, 12, 11, 5, 4, 0, set).
CSCG to AOP	CSCG to send new data to AOP (bits 13, 12, 11, 10, 5, 4, 3, 2, 0 set)

Table B-4
ACP to Communication System Control Group Command Words (via IOEXEC)


CSCG
CLEAR "STATUS SENT" FLAG IF INPUT BUFFER NOT EMPTY
DO WHILE INPUT BUFFER NOT EMPTY
NORMAL DATA XFER - CSCG TO AYK
T
F
CLEAR "DATA SENT" FLAG
NORMAL DATA XFER - AYK TO CSCG
T
F
READ IN DATA WORD BLOCK FROM INPUT BUFFER
INITIALIZE TERMINAL
T
F
DITS IN QUIESCENT STATE
OVERRIDE SELF-TEST IF IN PROGRESS
INITIATE PROCESSING
T
F
END OF QUIESCENT STATE
SET FLAG TO XMIT THE DATA WORDS
INITIATE SELF-TEST
T
F
INITIALIZE SELF-TEST COUNTER
SET "RECEIVE BUSY" BIT IN RT STATUS WORD
SET FLAG TO XMIT RT STATUS WORD
SET "INVALID DITS WORD" ERROR FLAG
END DO
GO TO 1


1
SELF-TEST IN PROGRESS
F
T
DECREMENT THE SELF-TEST COUNTER
COUNTER = 0
F
T
SELF-TEST ENDED. SET "RECEIVE BUSY BIT TO 0".
QUIESCENT STATE
T
F
NO SELF-TEST OR SELF-TEST JUST BE-GUN
F
T
UHF IN MANUAL MODE
F
T
READ IN MODE SETTING AND CHAN. SELEC. FROM DISCRETE
UPDATE D/L MODE SETTING SONO RCVRS, SIG. STRENGTH
OUTPUT SONO RCVR TUNING TO SONOBUOY
PP AND RT DATA AVBL FLAGS =
F
T
SET ERROR WORD BITS
STATUS SENT & DATA SENT FLAG = 0
F
T
SET ERROR WORD BITS
OUTPUT CSOG DATA WORD BLOCK AS REQUIRED
RETURN

APPENDIX C

MULTIFUNCTION CONTROL SET MODULE

The Multifunction Control Set (MFCS) module simulates the processing of discrete switch data between the AOP, and the Air Tactical Officer (ATO) and Sono Operator (SO) keysets. Switch closure discretes from the keysets are encoded and sent to the AOP, and keyswitch lighting data from the AOP is decoded and sent to the keysets via DADIOS. Communication between the keysets and the AOP, including output buffer error checks and the self-test, are covered in detail in section 3.

The module consists of four routines which are:

- MFCS - This is the driver routine for the module. Evertime the module is called each keyset is fully processed one at a time (first the ATO keyset, then the SO keyset). It scans the keyset's AOP input buffer for commands. When a "Normal Data Transfer" command is received MFCSNDT is called to complete the processing. The other commands (Initialize Terminal, Initiate Processing and Initiate Self-test) are processed as described in section 3. Completing the input buffer scan, routine MFCSPRC is called to continue normal processing if the quiescent and self-test states are inactive.
- MFCSNDT - This routine, called by MFCS, scans the data portion of a "normal data transfer" for switch lighting data. The routine has two distinct sections, one for each keyset:

  ATO - The ATO keyset section functions with only one set of out descretes (available inactive, and available active),

consequently, only the second half of the data words (available active) are processed. During the scan if the data bit is logic zero the out discrete is set to logic zero, available inactive state, and if logic one, available active state.

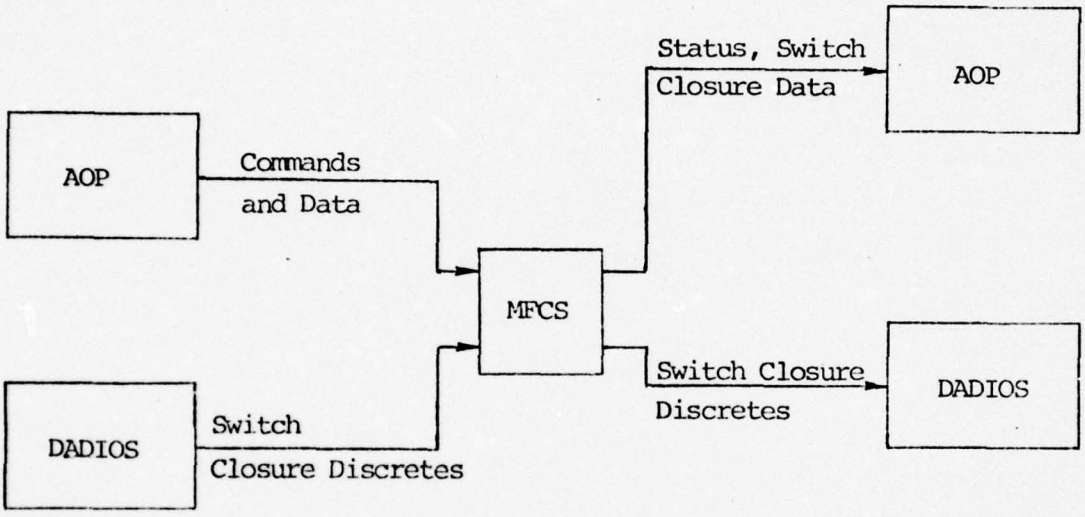
SO - The SO keyset section functions with two sets of outdiscretes - one set the available inactive state, the other the available active state - consequently both halves of the data words are processed. While processing the first half of the data words, a logic one places the outdiscrete to available inactive state; a logic zero does not change the state. During the processing of the second half of the data words, a logic one places the outdiscrete to available actual state; a logic zero does not change the state.

On completion of processing control returns to MFCS.

- MFCSPRC - This routine, called by MFCS, continues the module's processing. Change in "BIT" status processing is performed as described in section 3. MFCSIND is called to test for switch closures from the keyset. If there is data to be sent to the AOP and previous data has been processed (see section 3), the data is packed into the output buffer to the AOP via a call to PACKPP, output flag(s) are set and control returns to MFCS.
- MFCSIND - This routine, called by MFCSPRC, processes switch closure data from the keysets. The routine has two distinct portions, one for the ATO and the other for the SO keyset, since the indiscretes for each are unique. The indiscretes are scanned for up to two closures and the data is formatted into the output buffer to be sent to the AOP. On completion of its scan, control returns to MFCSPRC.

The MFCS module performs a subset of the functional requirements of the MFCS hardware. The disparity between the software and hardware is:

- The Master Clear function is not implemented.
- The manual "self-test" is not implemented.
- Due to hardware limitations in the LAMPS simulation hardware, not all of the keyswitches have been implemented.



MFCS INPUT AND OUTPUT TO AND FROM THE AOP AND DADIOS

Figure C-1

Word	Description
ATO/SO Keyset Switch Closure	Bits 15-9 Key Identification Code (unsigned)

MFCS TO AOP DATA FORMATS

Table C-1

Word	Description
Keyset Control Word 1	Bits 15-0 Switches 0 thru 15 (Logic 1 = Green)
Keyset Control Word 2	Bits 15-0 Switches 16 thru 31 (Logic 1 = Green)
Keyset Control Word 3	Bits 15-0 Switches 32 thru 47 (Logic 1 = Green)
Keyset Control Word 4	Bits 15-0 Switches 48 thru 63 (Logic 1 = Green)
Keyset Control Word 5	Bits 15-6 Switches 64 thru 73 (Logic 1 = Green)
Keyset Control Word 6	Bits 15-0 Switches 0 thru 15 (Logic 1 = Amber)
Keyset Control Word 7	Bits 15-0 Switches 16 thru 31 (Logic 1 = Amber)
Keyset Control Word 8	Bits 15-0 Switches 32 thru 47 (Logic 1 = Amber)
Keyset Control Word 9	Bits 15-0 Switches 48 thru 63 (Logic 1 = Amber)
Keyset Control Word 10	Bits 15-6 Switches 64 thru 73 (Logic 1 = Amber)

AOP TO MFCS DATA WORD FORMATS

Table C-2

SWITCH NUMBER	SWITCH DESCRIPTION
0	Numeric 0
1	Numeric 1
2	Numeric 2
3	Numeric 3
4	Numeric 4
5	Numeric 5
6	Numeric 6
7	Numeric 7
8	Numeric 8
9	Numeric 9
10	FLY TO
11	DATUM
12	ESM SCAN CNTRL (ESM Scan Control)
13	EXPND CIRCL (Expanding Circle)
14	MARK CURSR (Mark Cursor)
15	FIX
16	PRED POSIT (Predicted Position)
17	HOOK LAT/LONG (Hook Latitude/Longitude
18	SEND SYMBL (Send Symbol)
19	SEND POINT (Send Pointer)
20	DECR RANGE (Decrease Range)
21	INCR RANGE (Increase Range)
22	RECTR ON HELO (Recenter on Helo)
23	HELO CNTR STAB (Helo Centered Stabilization)
24	RECTR ON HOOK (Recenter on Hook)
25	TACT (Tactical)
26	TABLE
27	ESM
28	RADAR
29	IFF (Identification Friend or Foe)
30	RPT OTHER (Repeat Other)
31	NEXT PAGE
32	RADAR RCVR GAIN (Radar Receiver Gain)
33	PERST (Persistence)
34	RPM
35	DEST SYMB (Destroy Symbol)
36	HOOK VERFY (Hook Verify)
37	HELO CONTR (Helo Control)
38	CLEAR ALERT CUE
39	BACK SPACE
40	ENTER NO CHNG (Enter No Change)

ATO KEYSET SWITCH ALLOCATION

Table C-3 (Page 1 of 2)

SWITCH NUMBER	SWITCH DESCRIPTION
41	CURSR FROM HOOK (Cursor from Hook)
42	CURSR FROM HELO (Cursor from Helo)
43	ON TOP
44	TACAN CORR (TACAN Correction)
45	SHIP CORR (Ship Correction)
46	Spare
47	Spare
48	Spare
49	Spare
50	Spare
51	RANGE CIRCL (Range Circle)
52	SENSR HORIZ (Sensor Horizon)
53	ALL BUT
54	INHIB (Inhibit)
55	INVRS (Inverse)
56	TRACK CLASS
57	TRACK SYMB (Track Symbol)
58	REF MARK (Reference Mark)
59	SENSR CNTCT (Sensor Contact)
60	VISUL CNTCT (Visual Contact)
61	LOAD CMUX
62	LINK
63	INIT SYNC (Initiate Sync)
64	RCOVR DATA (Recover Data)
65	INIT HELO (Initialize Helo)
66	INSERT SONO (Insert Sonobuoy)
67	VHF
68	DEEP
69	SCUTL (Scuttle)
70	Spare
71	Spare
72	LAMP TEST
73	SELF TEST

ATO KEYSET SWITCH ALLOCATION

Table C-3 (Page 2 of 2)

COMMAND	DESCRIPTION
Mode/Discrete Data	
• Initialize Terminal	Module placed in quiescent state (bits 15, 0 set)
• Initiate Self-Test	Module placed in self-test state (bits 15, 1, 0 set)
• Initiate Processing	Modules internal processing commenced (bits 15, 2 set)
Normal Data Transfer	
• AOP to MFCS Transfer	Module to receive AOP data (bits 15, 5, 3, 1 set)
• MFCS to AOP Transfer	Module to transfer data to AOP (bits 15, 10, 5 set; bits 4 - 0 as required)

AOP to MFCS Commands (ATO)
Table C-4

COMMAND	DESCRIPTION
Mode/Discrete Data	
• Initialize Terminal	Module placed in quiescent state (bits 15, 14, 13, 0 set)
• Initiate Self-Test	Module placed in self-test state (bits 15, 14, 13, 1, 0 set)
• Initiate Processing	Modules internal processing commenced (bits 15, 14, 13, 2 set)
Normal Data Transfer	
• AOP to MFCS Transfer	Module to receive AOP data (bits 15, 14, 13, 5, 3, 1 set)
• MFCS to AOP Transfer	Module to transfer data to AOP (bits 15, 14, 13, 10, 1 set; bits 4 - 0 as required)

AOP to MFCS Commands (SO)
Table C-5

SWITCH NUMBER	SWITCH DESCRIPTION
0	Numeric 0
1	Numeric 1
2	Numeric 2
3	Numeric 3
4	Numeric 4
5	Numeric 5
6	Numeric 6
7	Numeric 7
8	Numeric 8
9	Numeric 9
10	Radar
11	IFF
12	ESM
13	RPT Other
14	Spare
15	ACSTS
16	ACSTS PROC OPTN
17	Tune RCVR
18	MAD
19	MAD Scale Chng
20	Table
21	Next Page
22	RPM

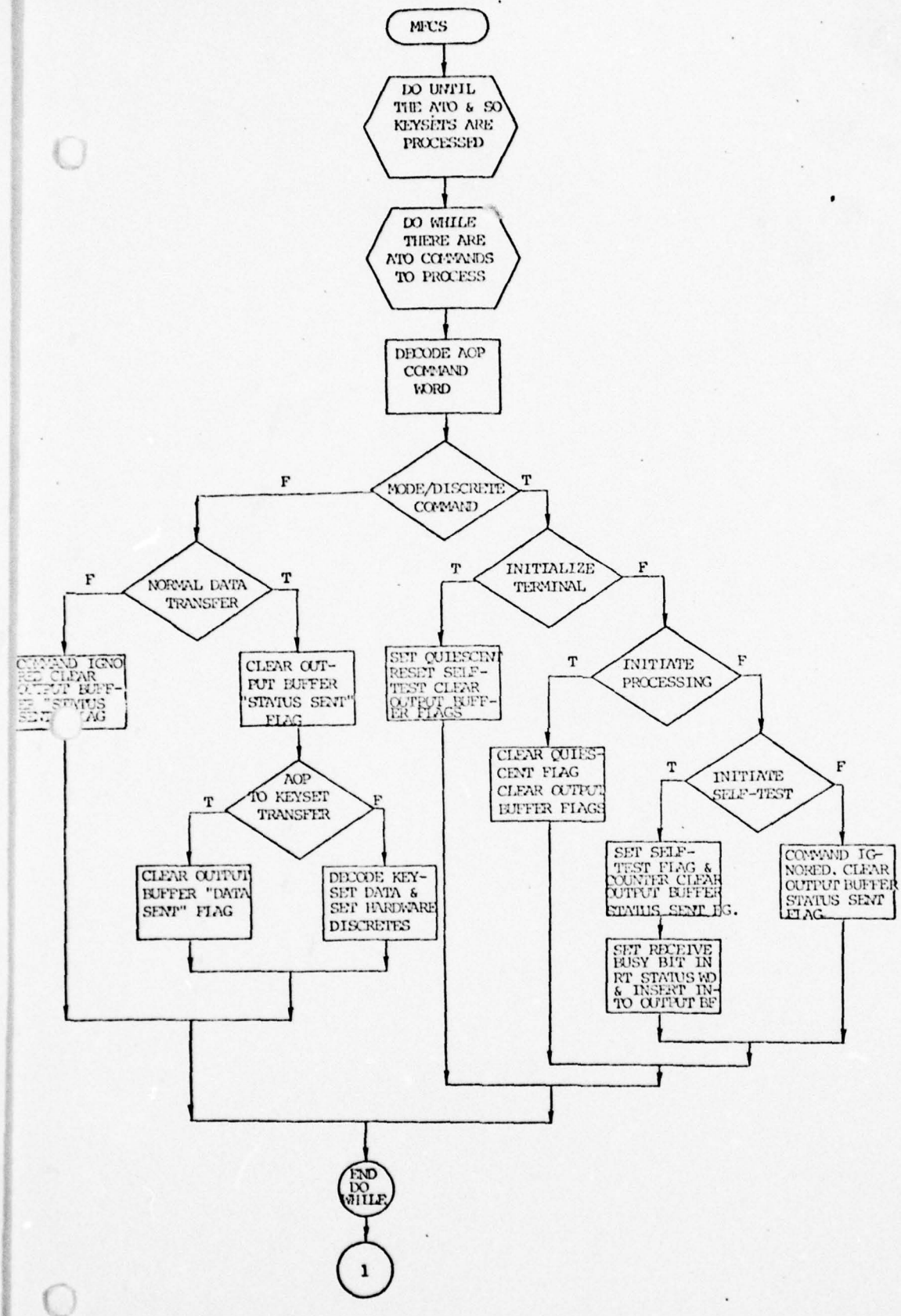
SO Keyset Switch Allocation
Table C-6

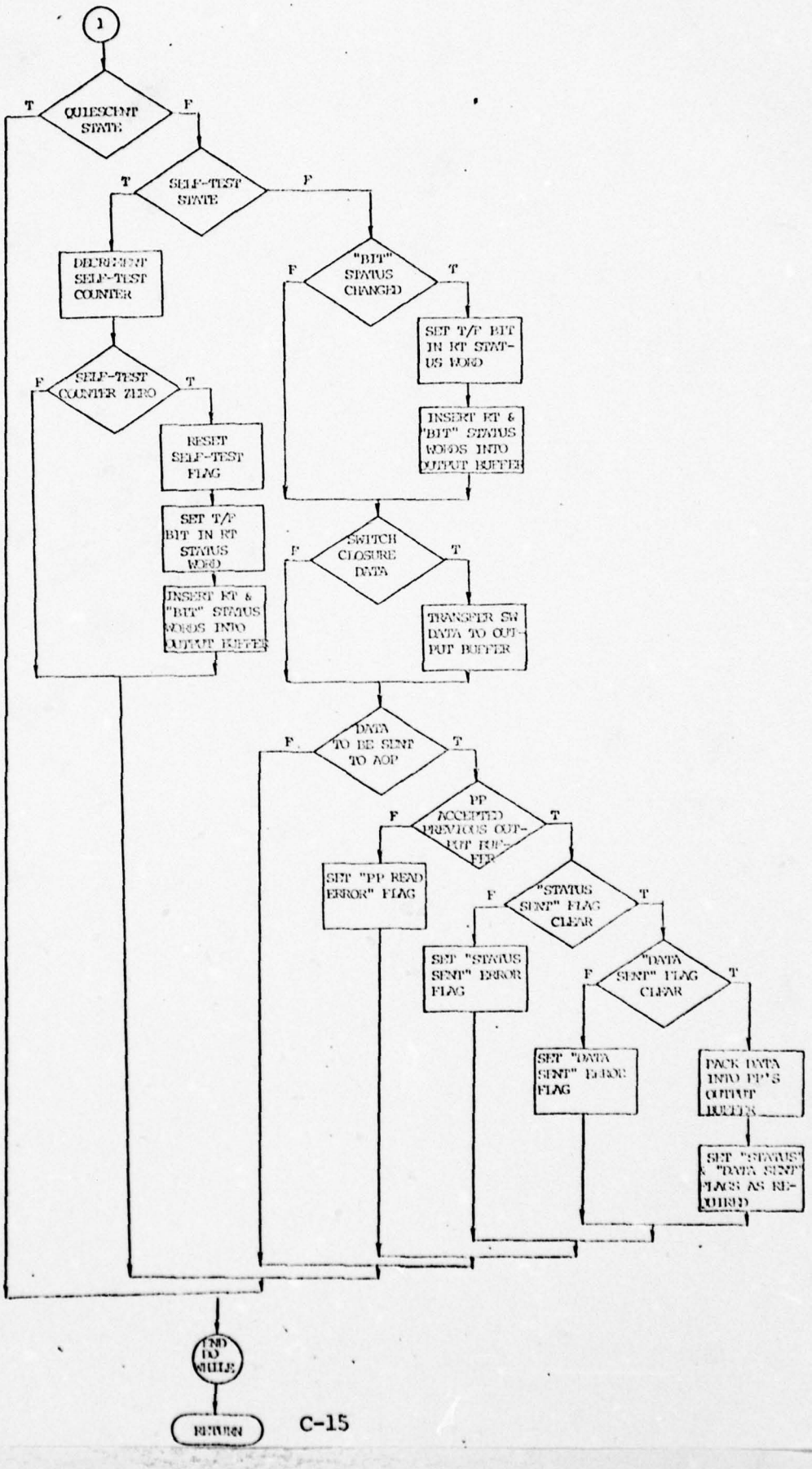
SWITCH NUMBER	SWITCH DESCRIPTION
23	PERST
24	CURSR from Helo
25	Cursr from Hook
26	Spare
27	Reset Chan
28	Chan A
29	Chan E
30	Disp Zone 1
31	Cursr Sel Audio
32	Mark ESM Line
33	Spare
34	Radar Rcvr Gain
35	Helo Cntr Stab
36	Rctr on Helo
37	Rctr on Hook
38	Spare
39	Intg Time
40	Chan B
41	Chan F
42	Displ Zone 2
43	Ping Rate
44	Hook Verify

SO Keyset Switch Allocation (Page 2 of 3)
Table C-6

SWITCH NUMBER	SWITCH DESCRIPTION
45	Dest Symb
46	Incr Range
47	Decr Range
48	New Track
49	Edit Track
50	Insp Chng
51	Vern
52	Chan C
53	Chan G
54	Disp Zone 3
55	Ping Type
56	Back Space
57	Enter No Change
58	Lamp Test
59	Self Test
60	Track Class
61	Spare
62	Chan D
63	Chan H
64	Displ Zone 4
65	Gram
66	Fix
67	Sensr Cntct
68	Clear Alert

SO Keyset Switch Allocation (Page 3 of 3)
Table C-6


MPCS
DO UNTIL THE ATO & SO KEYSETS ARE PROCESSED
DO WHILE THERE ARE ATO COMMANDS TO PROCESS
DECODE AOP COMMAND WORD
MODE/DISCRETE COMMAND
F
T
NORMAL DATA TRANSFER
INITIALIZE TERMINAL
COMMAND IGNORED CLEAR OUTPUT BUFFER "STATUS SENT" FLAG
CLEAR OUTPUT BUFFER "STATUS SENT" FLAG
SET QUIESCENT RESET SELF-TEST CLEAR OUTPUT BUFFER FLAGS
INITIATE PROCESSING
AOP TO KEYSET TRANSFER
CLEAR QUIESCENT FLAG CLEAR OUTPUT BUFFER FLAGS
INITIATE SELF-TEST
CLEAR OUTPUT BUFFER "DATA SENT" FLAG
DECODE KEYSET DATA & SET HARDWARE DISCRETES
SET SELF-TEST FLAG & COUNTER CLEAR OUTPUT BUFFER STATUS SENT FG.
COMMAND IGNORED, CLEAR OUTPUT BUFFER STATUS SENT FLAG
SET RECEIVE BUSY BIT IN RT STATUS WD & INSERT INTO OUTPUT BF
END DO WHILE
1


1
QUIESCENT STATE
T
F
SELF-TEST STATE
T
F
DECREMENT SELF-TEST COUNTER
"BIT" STATUS CHANGED
F
T
SET T/F BIT IN RT STATUS WORD
SELF-TEST COUNTER ZERO
F
T
INSERT RT & "BIT" STATUS WORDS INTO OUTPUT BUFFER
RESET SELF-TEST FLAG
SET T/F BIT IN RT STATUS WORD
INSERT RT & "BIT" STATUS WORDS INTO OUTPUT BUFFER
SWITCH CLOSURE DATA
F
T
TRANSFER SW DATA TO OUTPUT BUFFER
DATA TO BE SENT TO AOP
F
T
PP ACCEPTED PREVIOUS OUTPUT BUFFER
F
T
SET "PP READ ERROR" FLAG
"STATUS SENT" FLAG CLEAR
F
T
SET "STATUS SENT" ERROR FLAG
"DATA SENT" FLAG CLEAR
F
T
SET "DATA SENT" ERROR FLAG
PACK DATA INTO PP'S OUTPUT BUFFER
SET "STATUS" & "DATA SENT" FLAGS AS REQUIRED
END TO MULE
RETURN

APPENDIX D

MAD SIGNAL PROCESSOR MODULE

The MAD Signal Processor (MSP) module provides for the reception of simulated MAD data from the MADMOD subroutine and the processing of this data into event and contact messages for transmission to the AOP. In addition the module compares simulator values of helo altitude, heading, ground speed, roll, latitude and longitude with those received from the AOP and if the difference is greater than a prescribed amount, sets the appropriate bits within an error status word.

The module consists of three routines which are:

- MSP - This is the main routine of the module. It scans the input buffer from the AOP for command words (Initialize Terminal, Initiate Processing, Multi-message Transfer, Control Command Data Transfer, Initiate Self-test, Normal Data Transfer) and sets the proper operational state of the module. In the case of an initial program load (IPL) sequence, MSP ignores the IPL words, checks to see if the observed word count matches the multi-message word count specified in the control command word, and continues normal processing. Subroutine MSP then extracts the data word count field from the current command word and if the count is non-zero, the routine checks the identification field for each incoming data word and branches to individual sections of coding to check and compare helo data and output processing mode. Self-test processing for MSP is handled as described in Section 3. After checking to make sure the

initiate processing/mode discrete is not active, the lockon is set, and the trail is out, the routine handles output processing in the manner described in Section 3 with the output processing option as previously selected by the AOP. (The output processing options differ in the type of MAD data loaded into the output buffer.) Additionally, bit 2 in the BIT STATUS word is set if there was a previously detected multi-message error.

- MSPPACK - This routine packs a real value in a binary field of requested size when called by the output processing section of MSP.
- SETBIT - This routine sets a given bit within a specified word to 0 or 1 as desired.
- LOCATE - This routine translates latitude and longitude, given x and y values, into the grid reference point.

DATA WORD	DESCRIPTION
Helicopter Altitude	Bits 12 - 0 Binary value of helo altitude Bits 15 - 13 ID Code
Helicopter Ground Speed	Bits 12 - 0 Binary value of helo ground speed Bits 15 - 13 ID Code
Helicopter Heading	Bits 12 - 0 Binary value of helo magnetic heading Bits 15 - 13 ID/Code
Helicopter Roll	Bits 12 - 2 Binary value of helo roll in two's complement form Bits 15 - 13 ID Code
Total Mad Field (Word 1)	Bits 12 - 0 Total Field MAD sensor data Bits 15 - 13 ID Code
Total Mad Field (Word 2)	Bits 15 - 3 Total field MAD sensor data
Helicopter Heading	Bits 6 - 1 Latitude in BAMS Bits 12 - 7 Longitude in BAMS Bits 15 - 13 ID Code
Altitude Compensation	Bits 11 - 2 Binary value of altitude compensation signal Bit 12 - Sign bit Bits 15 - 13 ID Code
Processor Option Select	Bits 12 - 11 Processor Mode Bits 15 - 13 ID Code

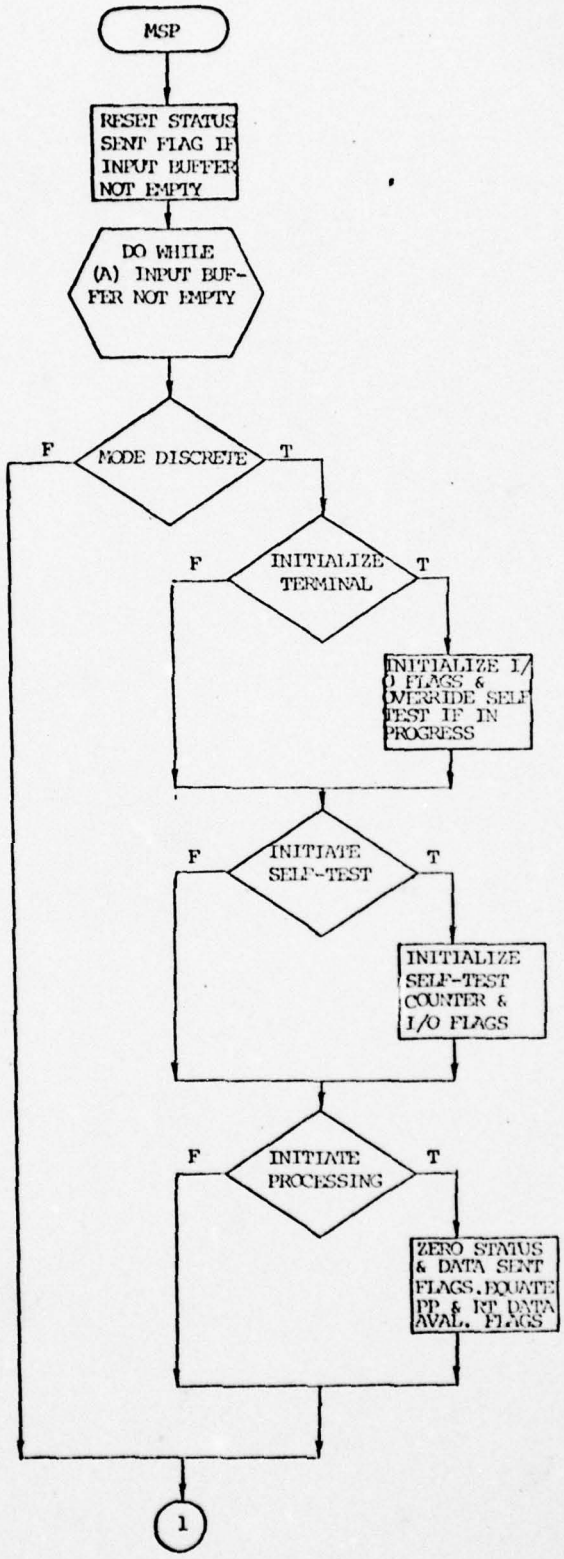
MAD Signal Processor Set to AOP Data Word Formats
Table D-1

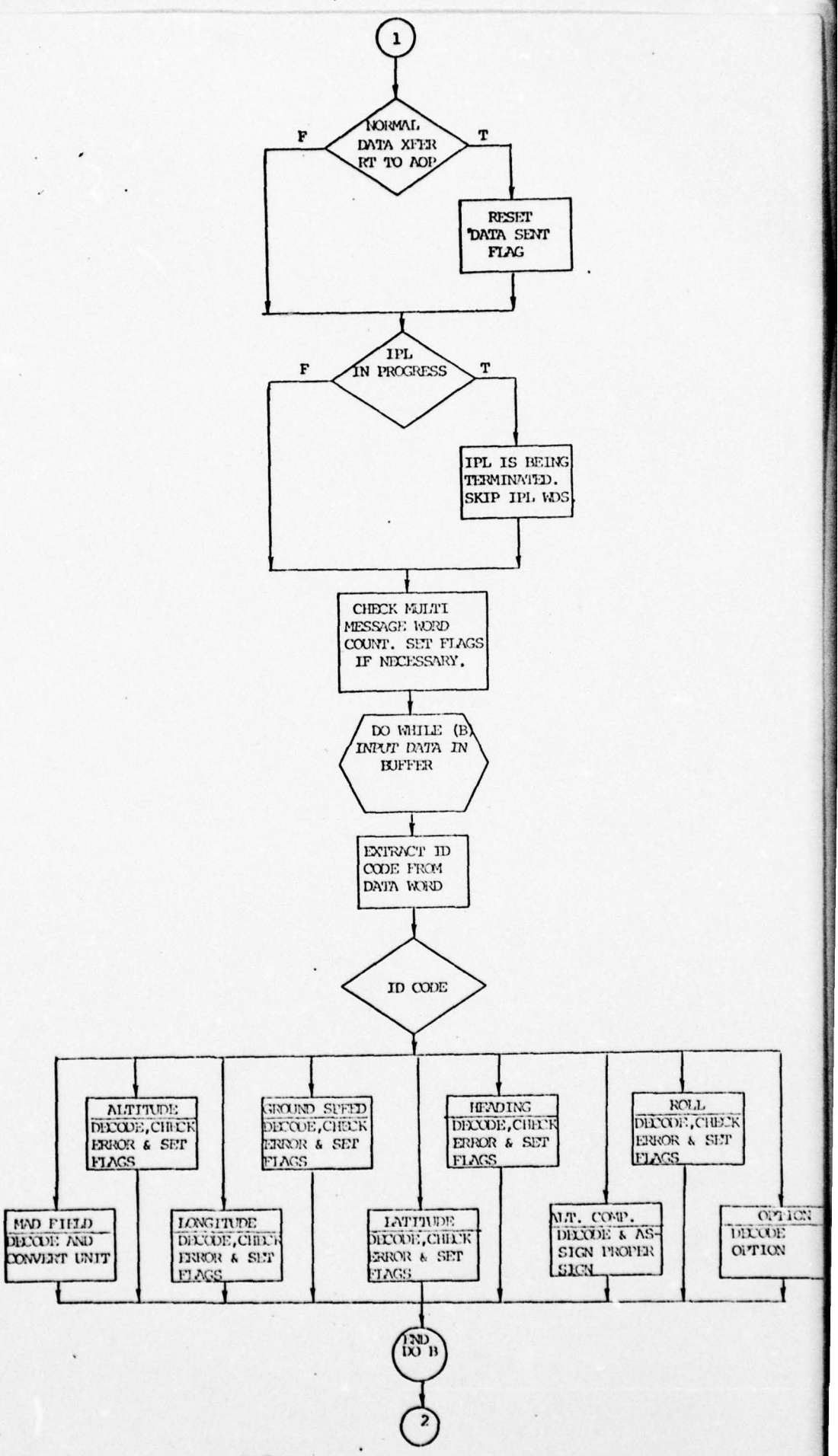
VARIABLE	DESCRIPTION
AMADDET	MAD processor storage array AMADDET (J,9) J = 1, 3 for three possible MAD marks (J,1) = Predetection Flag (J,8) = Predetection Confirmed Flag
IEVENT	Flag that predetection event is active
GRPLAT	Grid reference point latitude (degrees)
GRLONG	Grid reference point longitude (degrees)

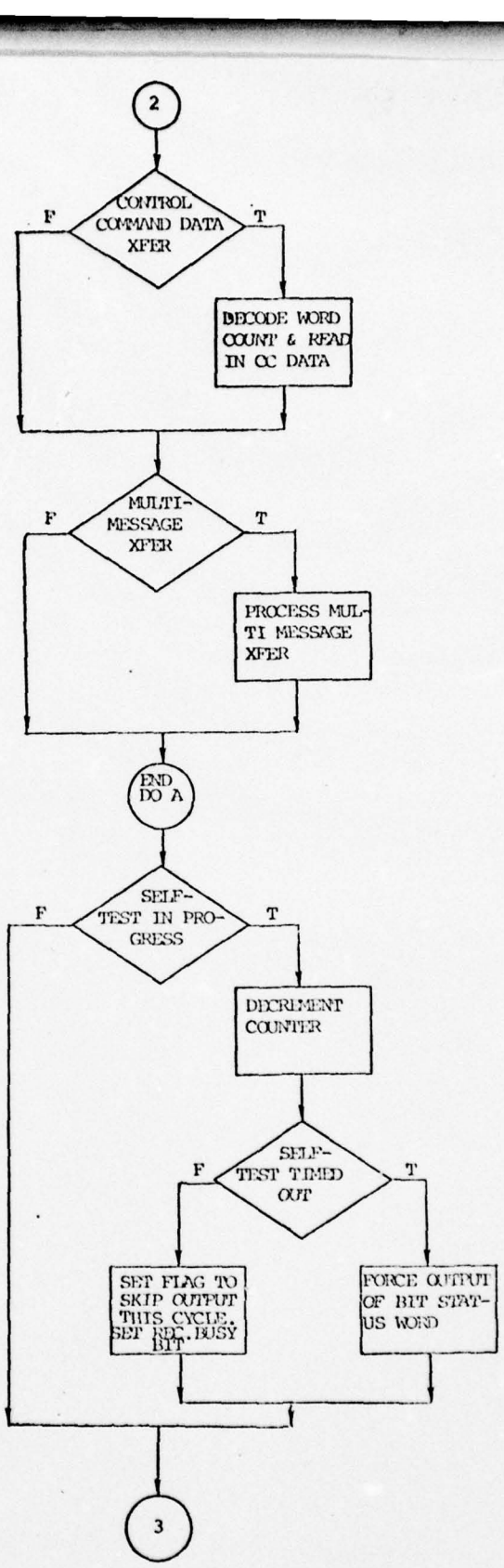
MAD Signal Processor Set/MADMOD Data Formats
Table D-2

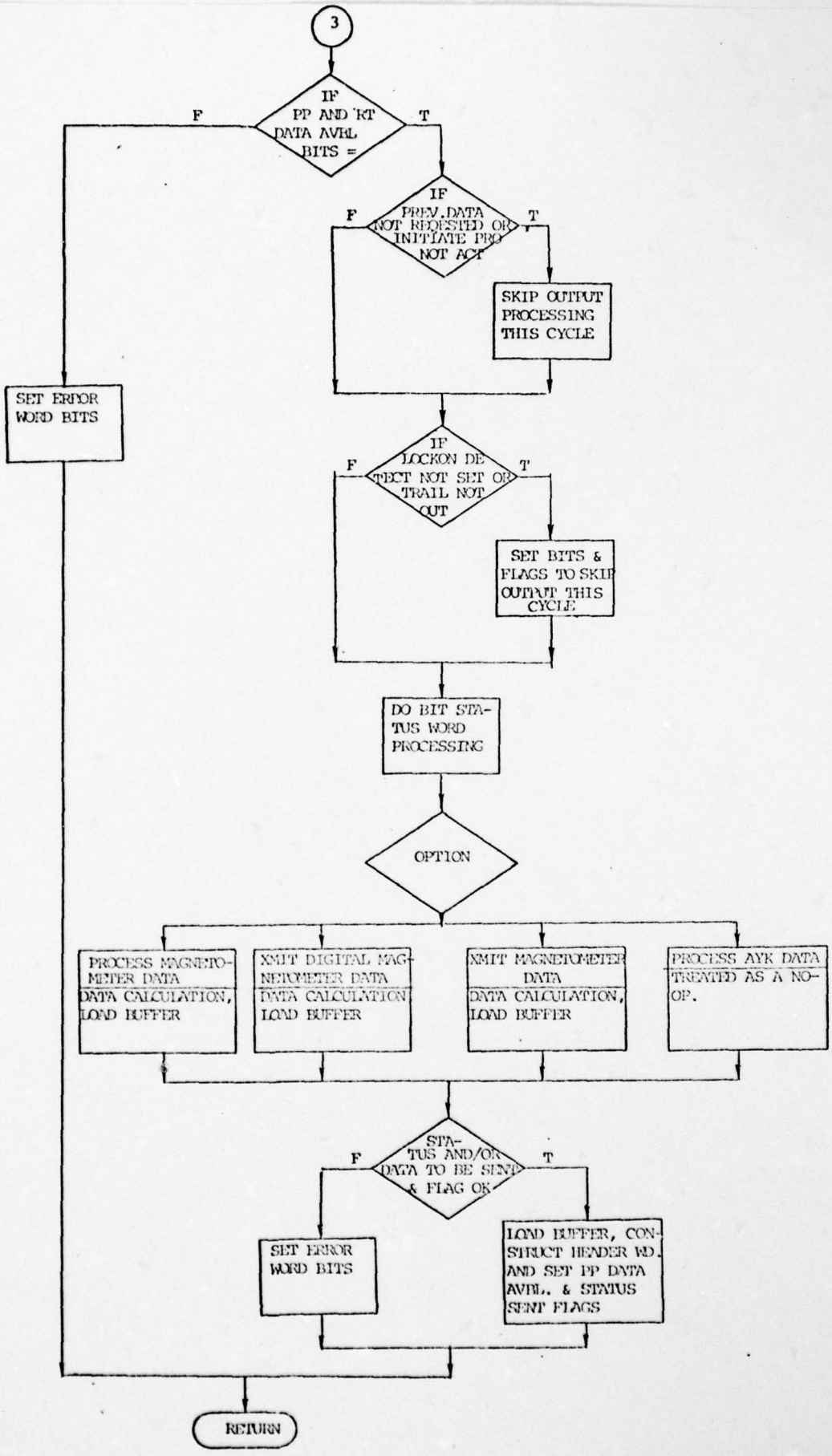
COMMAND	DESCRIPTION
Initialize Terminal	MSP forced into quiescent state, all processing halted (bits 14, 13, 11, 0 set)
Initiate Processing	MSP processing commenced (bits 14, 13, 11, 2, set)
Initiate Self-Test	MSP Built In Test (BIT) forced (bits 14, 13, 11, 1, 0 set)
Normal Data Transfer	
AOP to MSP	MSP to process new data from the AOP (bits 14, 13, 11, 5)
MSP to AOP	MSP to send new data to AOP (bits 14, 13, 11, 10, 5 set)
Multi-Message Transfer	MSP to process new data (bits 14, 13, 11, 7, 6 set)

AOP to MAD Signal Processor Set Command Words
Table D-3


MSP
RESET STATUS SENT FLAG IF INPUT BUFFER NOT EMPTY
DO WHILE (A) INPUT BUFFER NOT EMPTY
MODE DISCRETE
F
T
INITIALIZE TERMINAL
F
T
INITIALIZE I/O FLAGS & OVERRIDE SELF TEST IF IN PROGRESS
INITIATE SELF-TEST
F
T
INITIALIZE SELF-TEST COUNTER & I/O FLAGS
INITIATE PROCESSING
F
T
ZERO STATUS & DATA SENT FLAGS. EQUATE PP & RT DATA AVAL. FLAGS
1


1
NORMAL DATA XFER RT TO AOP
F
T
RESET "DATA SENT FLAG
IPL IN PROGRESS
F
T
IPL IS BEING TERMINATED. SKIP IPL WDS.
CHECK MULTI MESSAGE WORD COUNT. SET FLAGS IF NECESSARY.
DO WHILE (B) INPUT DATA IN BUFFER
EXTRACT ID CODE FROM DATA WORD
ID CODE
ALTITUDE DECODE, CHECK ERROR & SET FLAGS
GROUND SPEED DECODE, CHECK ERROR & SET FLAGS
HEADING DECODE, CHECK ERROR & SET FLAGS
ROLL DECODE, CHECK ERROR & SET FLAGS
MAD FIELD DECODE AND CONVERT UNIT
LONGITUDE DECODE, CHECK ERROR & SET FLAGS
LATTITUDE DECODE, CHECK ERROR & SET FLAGS
ALT. COMP. DECODE & ASSIGN PROPER SIGN
OPTION DECODE OPTION
END DO B
2


2
CONTROL COMMAND DATA XFER
F
T
DECODE WORD COUNT & READ IN CC DATA
MULTI-MESSAGE XFER
F
T
PROCESS MULTI MESSAGE XFER
END DO A
SELF-TEST IN PROGRESS
F
T
DECREMENT COUNTER
SELF-TEST TIMED OUT
F
T
SET FLAG TO SKIP OUTPUT THIS CYCLE, SET RBE.BUSY BIT
FORCE OUTPUT OF BIT STATUS WORD
3


3
IF PP AND RT DATA AVBL BITS =
F
T
IF PREV. DATA NOT REQESTED OR INITIATE PRO NOT ACT
F
T
SKIP OUTPUT PROCESSING THIS CYCLE
SET ERROR WORD BITS
IF LOCKON DETECT NOT SET OR TRAIL NOT OUT
F
T
SET BITS & FLAGS TO SKIP OUTPUT THIS CYCLE
DO BIT STATUS WORD PROCESSING
OPTION
PROCESS MAGNETOMETER DATA
DATA CALCULATION, LOAD BUFFER
XMIT DIGITAL MAGNETOMETER DATA
DATA CALCULATION LOAD BUFFER
XMIT MAGNETOMETER DATA
DATA CALCULATION, LOAD BUFFER
PROCESS AYK DATA
TREATED AS A NO-OP.
STATUS AND/OR DATA TO BE SENT & FLAG OK
F
T
SET ERROR WORD BITS
LOAD BUFFER, CONSTRUCT HEADER WD. AND SET PP DATA AVBL. & STATUS SENT FLAGS
RETURN

APPENDIX E

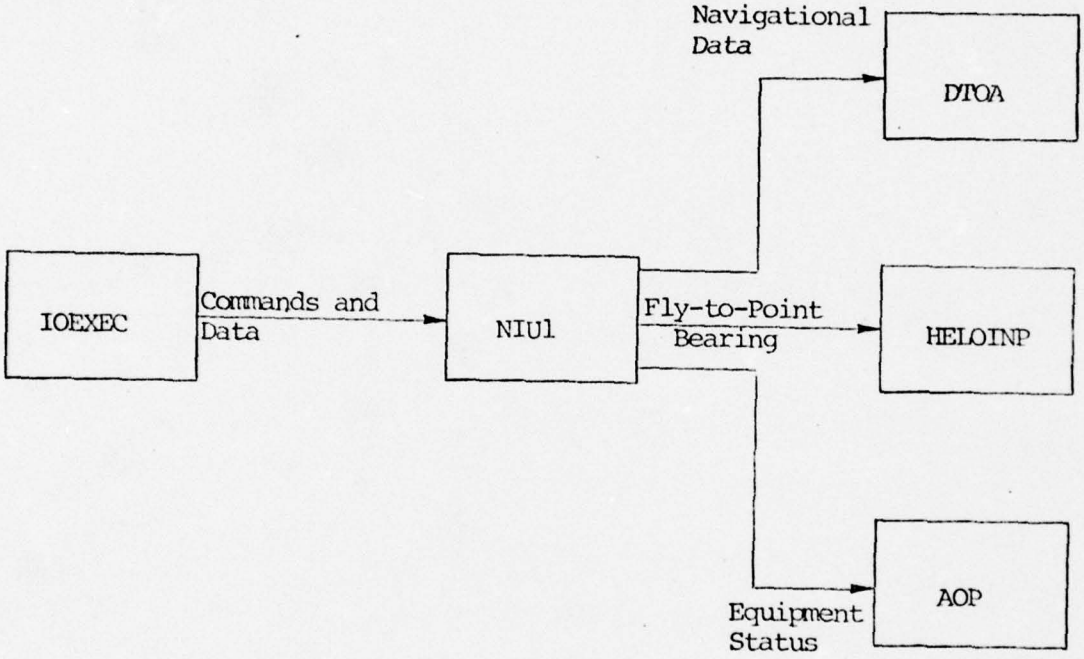
NAVIGATION INTERFACE UNIT MODULE

The Navigation Interface Unit (NIU) Module performs the simulated control of AOP commands, updating of fly-to-point bearings and transfer of AOP navigation data to the DTOA module. The module consists of the following routine:

- NIU1 - The routine scans the input buffer from the AOP for command words (Normal Data Transfer, Initialize Terminal, Initiate Processing, or Initiate Self-Test). When a normal Data Transfer (AOP sending data) is received, the navigational data (tactical range and bearing, drift angle and heading) are transferred to DTOA and the fly-to-point bearing is decoded for the Helo routines. The other commands, internal operation states and output of status and data to the AOP are processed as described in Section 3.

The NIU module performs a subset of the functional requirements of the NIU hardware is:

- The Master Clear function is not implemented.
- The manual "self-test" is not implemented.



NIU Inputs and Outputs to and from the AOP and Other Modules
Figure E-1

WORD	DESCRIPTION
Equipment Status Word 1	Bit 13 - PLT Selected
Equipment Status Word 2	Bit 15 - Pilot TACAN Selected Bit 14 - Pilot Computer Selected Bit 12 - ATO TACAN Selected Bit 11 - ATO Computer Selected

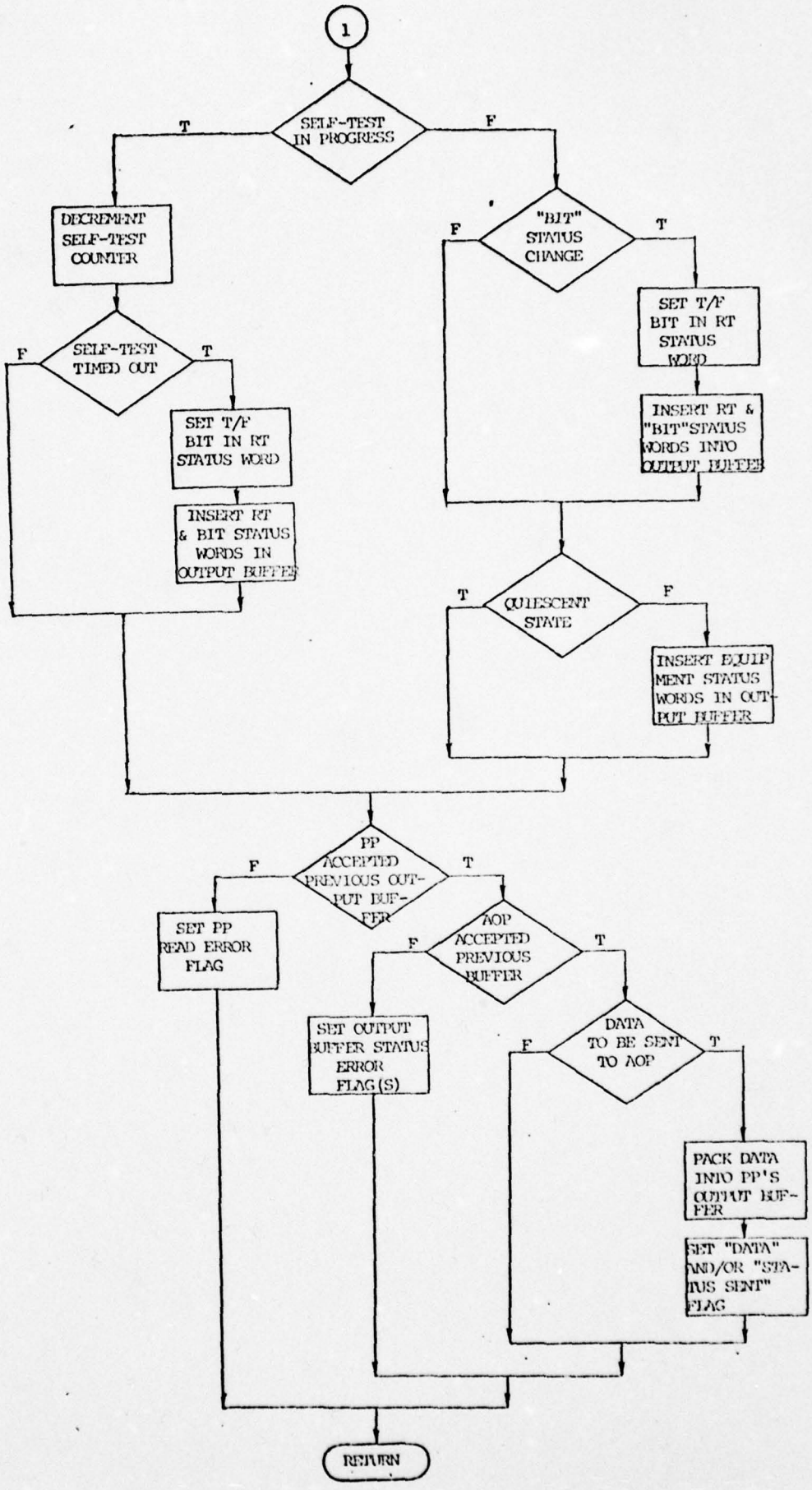
Navigational Interface Unit to AOP Status Formats
Table E-1

WORD	DESCRIPTION
FTP Tactical Range	Bits 15 - 1 Tactical Range, unsigned Bit 0 Range Flag
FTP Tactical Bearing	Bits 15 - 1 Tactical Bearing, unsigned Bit 0 Navigation Flag
Drift Angle	Bits 15 - 1 Drift Angle, unsigned
Pilot's True Heading	Bits 15 - 1 Heading, unsigned
ATO's True Heading	Bits 15 - 1 Heading, unsigned

IOEXEC to Navigation Interface Unit Data Formats
Table E-2

COMMAND	DESCRIPTION
Mode/Discrete Data	
• Initialize Terminal	Module placed in quiescent state (bits 14, 13, 12, 0 set)
• Initiate Self-Test	Module placed in self-test state (bits 14, 13, 12, 1, 0 set)
• Initiate Processing	Modules internal processing commenced (bits 14, 13, 12, 2 set)
Normal Data Transfer	
• AOP to NIU Transfer	Module to receive AOP data (bits 14, 13, 12, 5, 1 set)
• NIU to AOP Transfer	Module to transfer data to AOP (bits 14, 13, 12, 5, 2, 1 set)

AOP to NIU Commands
Table E-3


1
SELF-TEST IN PROGRESS
T
F
DECREMENT SELF-TEST COUNTER
SELF-TEST TIMED OUT
F
T
SET T/F BIT IN RT STATUS WORD
INSERT RT & BIT STATUS WORDS IN OUTPUT BUFFER
"BIT" STATUS CHANGE
F
T
SET T/F BIT IN RT STATUS WORD
INSERT RT & "BIT" STATUS WORDS INTO OUTPUT BUFFER
QUIESCENT STATE
T
F
INSERT EQUIPMENT STATUS WORDS IN OUTPUT BUFFER
PP ACCEPTED PREVIOUS OUTPUT BUFFER
F
T
SET PP READ ERROR FLAG
AOP ACCEPTED PREVIOUS BUFFER
F
T
SET OUTPUT BUFFER STATUS ERROR FLAG(S)
DATA TO BE SENT TO AOP
F
T
PACK DATA INTO PP'S OUTPUT BUFFER
SET "DATA" AND/OR "STATUS SENT" FLAG
RETURN

APPENDIX F

ORDNANCE LAUNCH CONTROL SET MODULE

The Ordnance Launch Control Set (OLCS) software module provides for simulated sonobuoy and torpedo select and launch commands via a software command from the AOP and manual keyset inputs. In addition to the normal sonobuoy processing, OLCS provides the rest of the simulator with splash point and water entry time calculations upon the launch of a sonobuoy or torpedo.

The module consists of ten routines which are:

- OLCS - This is the main routine which directs and controls all functions performed in the OLCS module. It scans the input buffer from the AOP for command words (Normal Data Transfer, Control Command Data Transfer, Initialize Terminal, Initiate Processing, or Initiate Self-test), sets the appropriate operational states of the module, and sets an error flag should an invalid command word appear in the input stream. Self-test processing is then handled as described in section 3. The OLCS routine, through the use of several additional subroutines (subroutines UDOASP, UDOH, SPLASH, WET, TSPLASH, and TWET) handles those functions which are unique to the OLCS module. These are described below. Finally, the routine handles output processing as described in section 3.
- UDOASP - Subroutine UDOASP updates manual keyset inputs (select/launch mode, torpedo and master arm, torpedo and sonobuoy launch, and manual sonobuoy select) from the simulated Ordnance Arm and Select Panel, resets data word bits for transmission to the AOP, and sets flags to indicate the necessity of a splash point or water entry time calculation.

- UDOH - Subroutine UDOH contains a counter which is initialized when a sonobuoy or torpedo launch occurs. This counter is then decremented on each subsequent OLCS call until reaching a zero value and a torpedo or sonobuoy away signal is generated for the AOP in the form of a bit set in an outgoing data word.
- SPLASH - Subroutine SPLASH makes the splash point calculation for a launched sonobuoy and outputs this information to sonobuoy. The routine also sets an "in water" flag and resets bits in the OLCS data words to indicate a launch.
- WET - This subroutine calculates water entry times for launched sonobuoys for the sonobuoy routine (an existing routine currently in the simulator that performs all sonobuoy simulation).
- CONTROL - In the event of a Control Command Data Transfer, CONTROL extracts the sonobuoy launch information from the CC Data word and processes this data by setting flags, recording the selected chute number and initializing the sonobuoy away signal counter in the event of a sonobuoy auto launch command.
- TSPLASH - Subroutine TSPLASH makes splash point calculations for launched torpedoes for the sonobuoy routine and sets a torpedo active flag.
- TWET - This subroutine makes water entry time calculations for launched torpedoes for the sonobuoy routine.
- SETABIT - Subroutine SETABIT sets a given bit within a specified word to 0 or 1 as requested.
- READBIT - Subroutine READBIT right justifies and returns the value of a bit within a given word.

The OLCS module performs a subset of the functional requirements of the OLCS hardware. The disparity between the hardware and the software is:

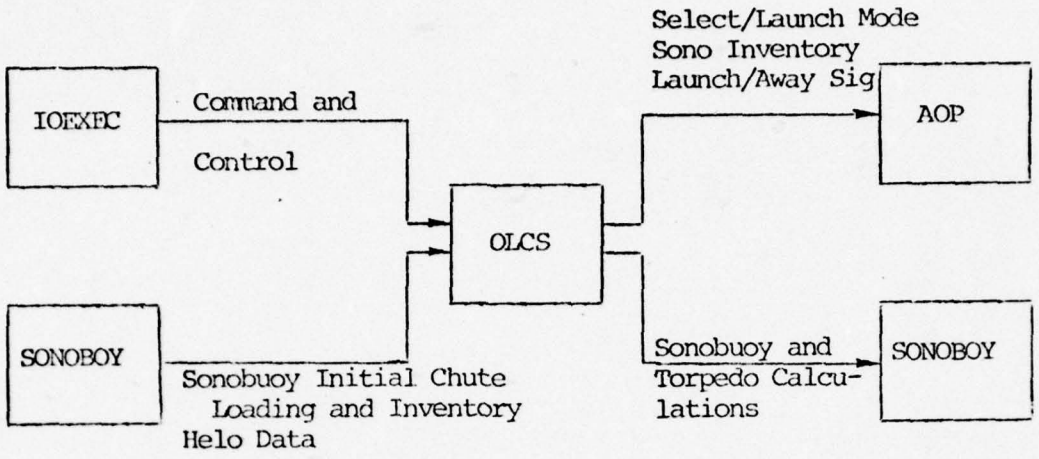
- The Master Clear function is not implemented.
- The manual "self-test" is not implemented.
- Due to simulator hardware limitations only buoys 1-19 can be manual/auto launched. Buoys 20-25 must be auto launched.
- Emergency jettison of sonobuoys is not implemented.
- Weight on wheels interlock is not implemented.
- Due to simulator hardware restrictions, sonobuoy manual/auto - select/launch commands cannot be mixed.
- Provision for left/right, search depth, mode/ceiling, course setting and emergency jettison of torpedoes is not implemented.

DATA WORD	DESCRIPTION
Sonobuoy Inventory Status Word 1	Bit 15 Sono. Auto Select Mode Bit 14 Sono. Auto Launch Mode Bit 13 Sono. Launch Signal Bit 12 Sono. Away Signal Bit 11 Torp. Launch Signal Bit 10 Torp. Away Signal Bits 8 - 0 Sono. Chutes 1 - 9 (loaded or unloaded)
Sonobuoy Inventory Status Word 2	Bits 15 - 0 Sono Chutes 10 to 25 (loaded or unloaded)

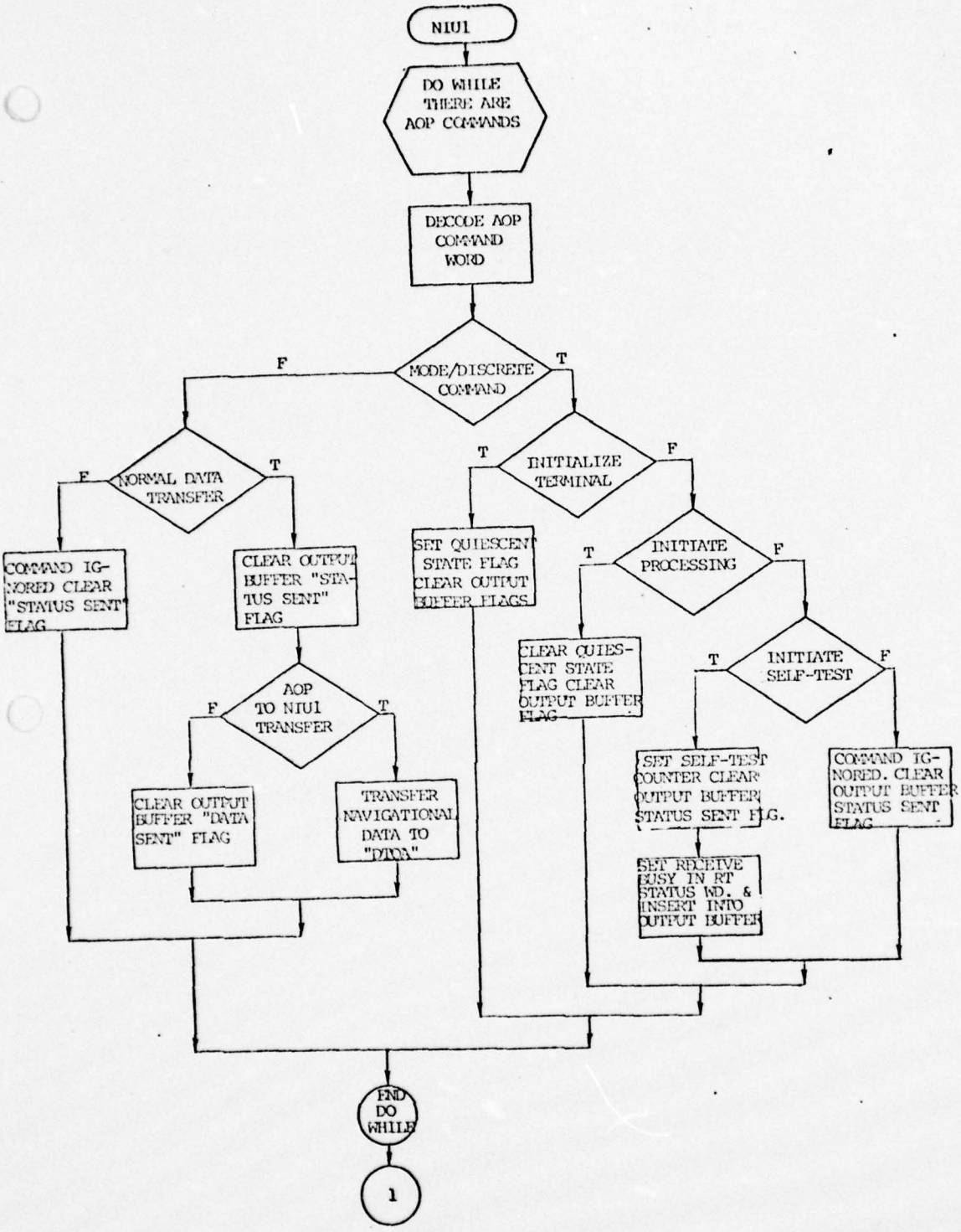
Ordnance Launch Control Set to AOP Data Formats
Table G-1

DATA WORD	DESCRIPTION
Sonobuoy Inventory Status Word 1	Bit 15 Sono. Auto Select Mode Bit 14 Sono. Auto Launch Mode Bit 13 Sono. Launch Signal Bit 12 Sono. Away Signal Bit 11 Torp. Launch Signal Bit 10 Torp. Away Signal Bits 8 - 0 Sono. Chutes 1 - 9 (loaded or unloaded)
Sonobuoy Inventory Status Word 2	Bits 15 - 0 Sono Chutes 10 to 25 (loaded or unloaded)

Ordnance Launch Control Set to AOP Data Formats
Table G-1



Inputs and Outputs to and from the AOP and other Modules
Figure G-1


NIU1
DO WHILE THERE ARE AOP COMMANDS
DECODE AOP COMMAND WORD
MODE/DISCRETE COMMAND
F
T
NORMAL DATA TRANSFER
F
T
COMMAND IGNORED CLEAR "STATUS SENT" FLAG
CLEAR OUTPUT BUFFER "STATUS SENT" FLAG
AOP TO NIU1 TRANSFER
F
T
CLEAR OUTPUT BUFFER "DATA SENT" FLAG
TRANSFER NAVIGATIONAL DATA TO "DTDA"
INITIALIZE TERMINAL
T
F
SET QUIESCENT STATE FLAG CLEAR OUTPUT BUFFER FLAGS
INITIATE PROCESSING
T
F
CLEAR QUIESCENT STATE FLAG CLEAR OUTPUT BUFFER FLAG
INITIATE SELF-TEST
T
F
SET SELF-TEST COUNTER CLEAR OUTPUT BUFFER STATUS SENT FLG.
SET RECEIVE BUSY IN RT STATUS WD. & INSERT INTO OUTPUT BUFFER
COMMAND IGNORED. CLEAR OUTPUT BUFFER STATUS SENT FLAG
END DO WHILE
1

VARIABLE	DESCRIPTION
HELO(1)	Helo heading in radians
HELO(2)	Cosine of helo heading
HELO(3)	Sine of helo heading
HELO (13)	Helo x-coordinate in feet
HELO (14)	Helo Y-coordinate in feet
HELO (15)	Helo altitude in feet
HELO (21)	Helo air speed in ft/sec.
WIND (1)	Wind direction in angular degrees
WIND (2)	Wind speed in ft/sec.
TIME	Mission time in seconds
BUOYIC (1,J)	Buoy-type initial chute loading (J=1,25)

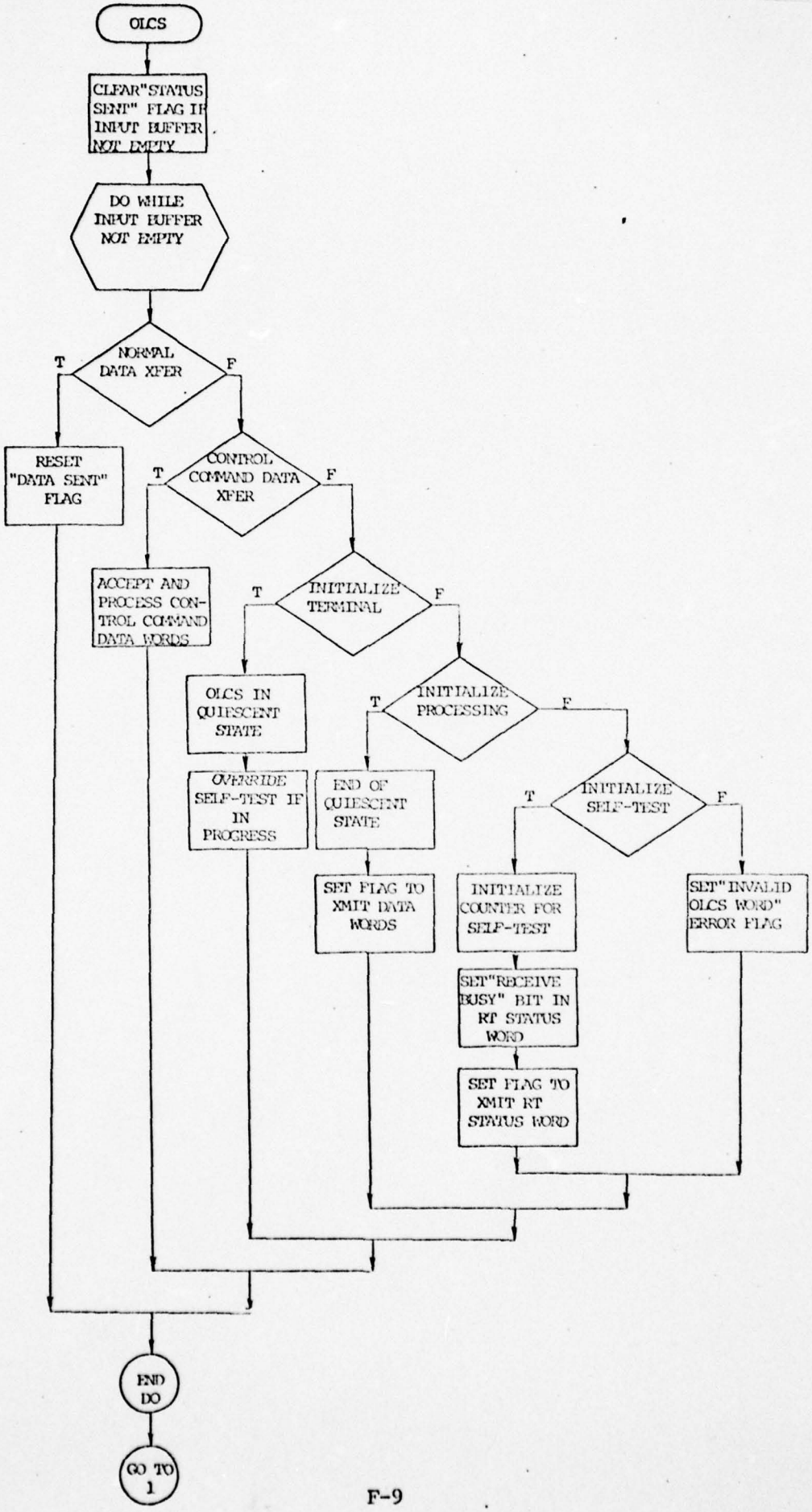
Sonoboy to Ordnance Launch Control Set Formats
Table G-2

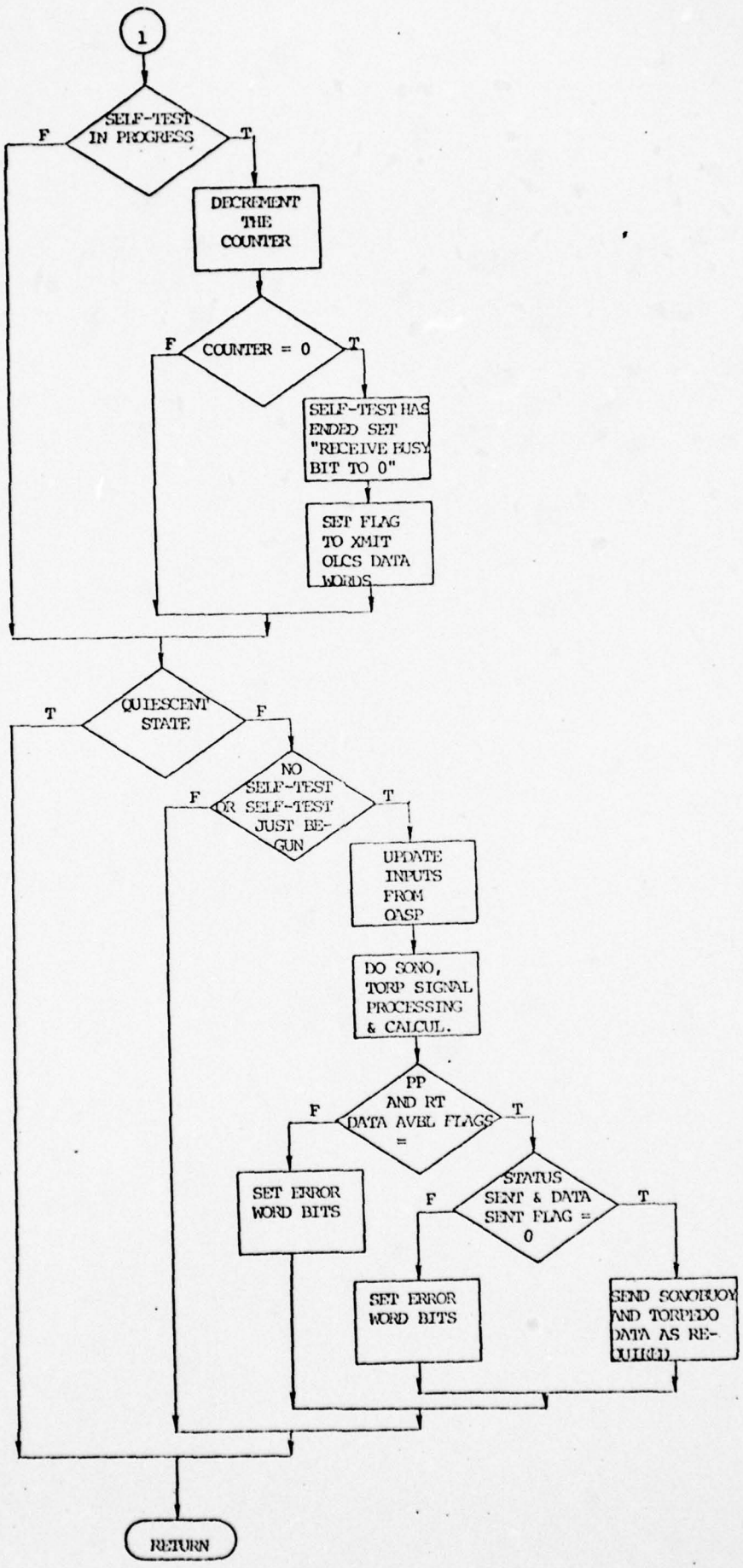
VARIABLE	DESCRIPTION
TOREP (1, I)	Torpedo X position in feet (I = 1,2)
TORPED (2, I)	Torpedo y position in feet (I = 1, 2)
TORPED (3, I)	Water entry time for torpedo (I = 1,2)
BUOYRW (2, I)	Sonobuoy x-coordinate (I = 1, 32)
BUOYRW (3, I)	Sonobuoy y-coordinate (I = 1, 32)
BUOYRW (4, I)	In water flag (I = 1, 32)
BUOYRW (11,I)	Water entry time for sonobuoy (I = 1, 32)
ITORDS	Torpedo symbol active flag

Ordnance Launch Control Set to Sonoboy Formats
Table G-3

COMMAND	DESCRIPTION
Initialize Terminal	OLCS forced into quiescent state, all processing halted (bits 15, 14, 11, 0 set)
Initiate Processing	OLCS processing commenced (bits 15, 14, 11, 2 set)
Initiate Self-Test	OLCS Built in Test (BIT) forced (bits 15, 14, 11, 1, 0 set).
Normal Data Transfer OLCS to AOP	OLCS to send new data to AOP (bits 15, 14, 11, 5, 1 set)
Control Command Data Transfer	OLCS to process new control command data (bits 15, 14, 11, 7, 5, 0, set).
CCDW1	Bits 11 - 15 auto sono select commands.
CCDW2	Bits 5 - 9 auto launch commands.

Table G-4
AOP to Ordnance Launch Control Set Command Words (via IOEXEC)


OLCS
CLEAR "STATUS SENT" FLAG IF INPUT BUFFER NOT EMPTY
DO WHILE INPUT BUFFER NOT EMPTY
NORMAL DATA XFER
T
F
RESET "DATA SENT" FLAG
CONTROL COMMAND DATA XFER
T
F
ACCEPT AND PROCESS CONTROL COMMAND DATA WORDS
INITIALIZE TERMINAL
T
F
OLCS IN QUIESCENT STATE
OVERRIDE SELF-TEST IF IN PROGRESS
INITIALIZE PROCESSING
T
F
END OF QUIESCENT STATE
SET FLAG TO XMIT DATA WORDS
INITIALIZE SELF-TEST
T
F
INITIALIZE COUNTER FOR SELF-TEST
SET "RECEIVE BUSY" BIT IN RT STATUS WORD
SET FLAG TO XMIT RT STATUS WORD
SET "INVALID OLCS WORD" ERROR FLAG
END DO
GO TO 1

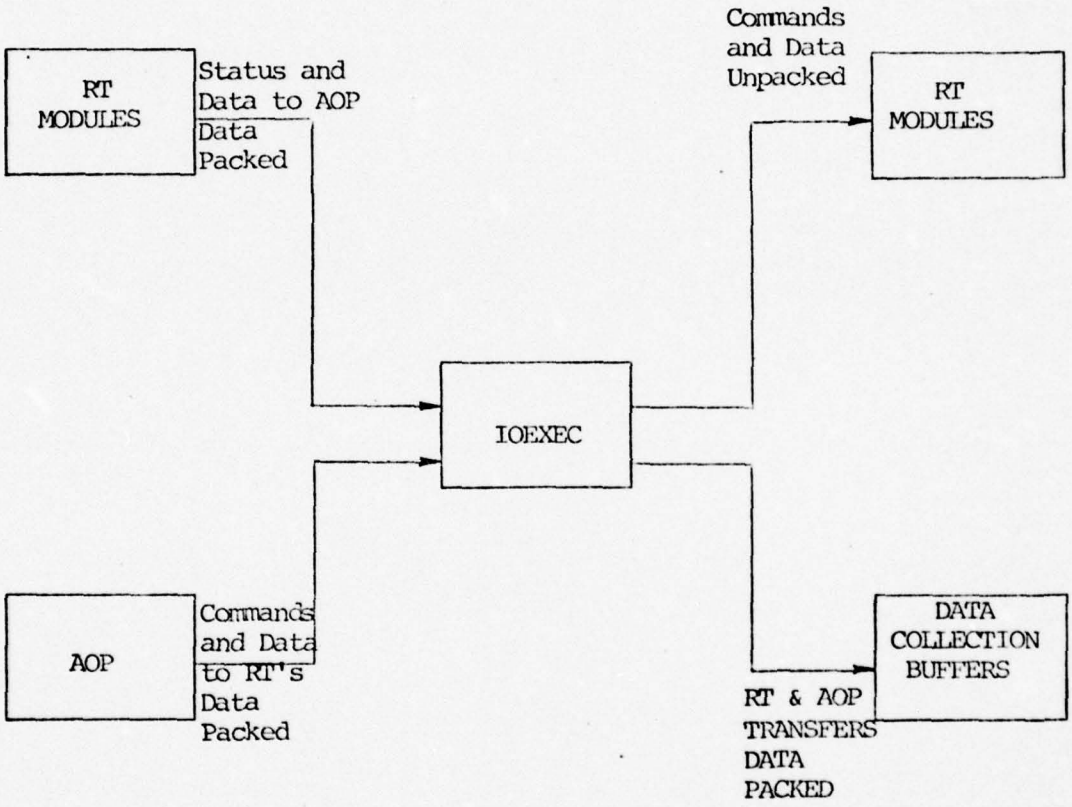

1
SELF-TEST IN PROGRESS
F
T
DECREMENT THE COUNTER
COUNTER = 0
F
T
SELF-TEST HAS ENDED SET "RECEIVE BUSY BIT TO 0"
SET FLAG TO XMIT OLCS DATA WORDS
QUIESCENT STATE
T
F
NO SELF-TEST OR SELF-TEST JUST BE-GUN
F
T
UPDATE INPUTS FROM OASP
DO SONO, TORP SIGNAL PROCESSING & CALCUL.
PP AND RT DATA AVBL FLAGS =
F
T
SET ERROR WORD BITS
STATUS SENT & DATA SENT FLAG = 0
F
T
SET ERROR WORD BITS
SEND SONOBUOY AND TORPEDO DATA AS RE-QUIRED
RETURN

APPENDIX G

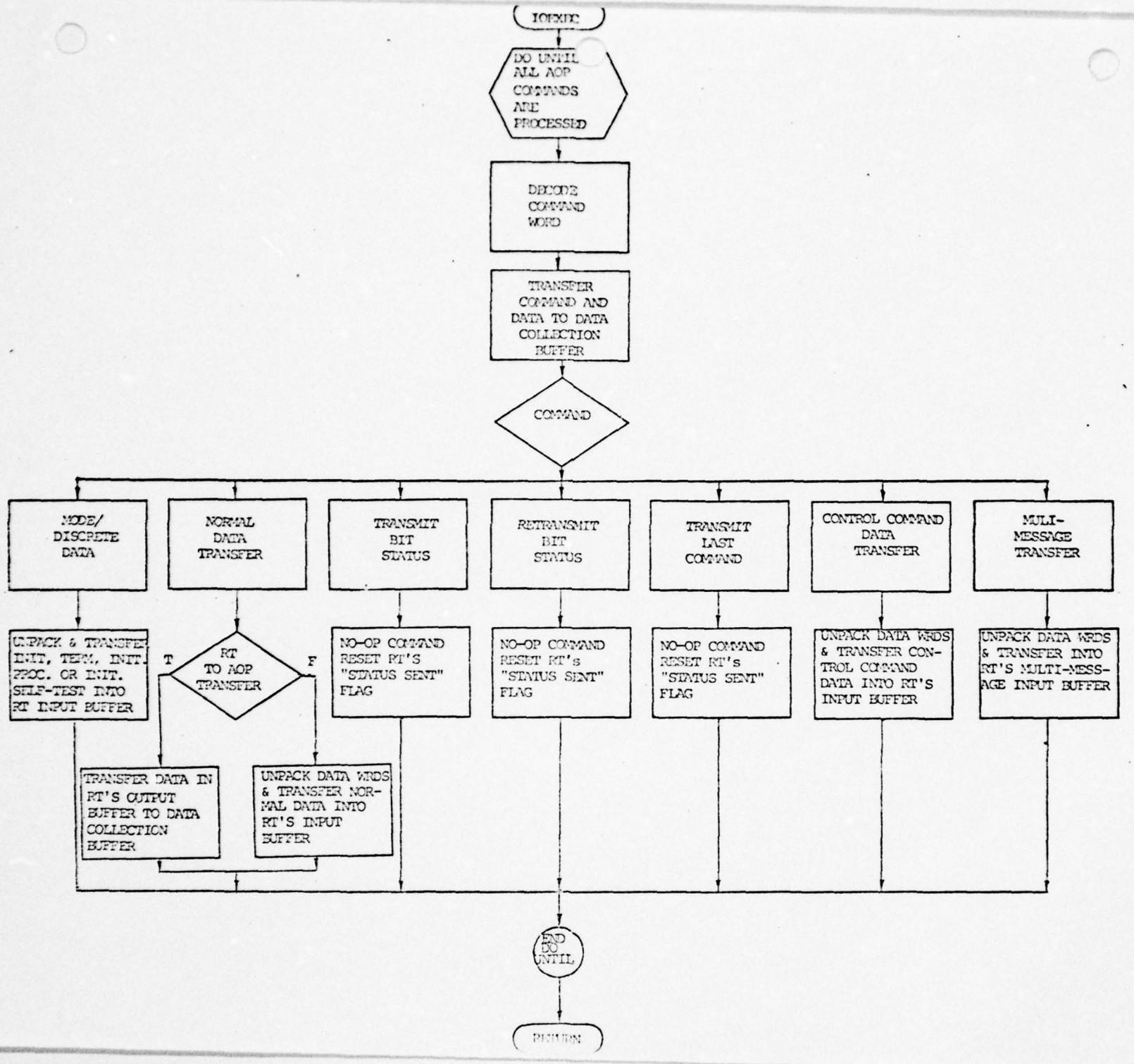
INPUT/OUTPUT EXECUTIVE AND DATA COLLECTION MODULE

The Input/Output Executive and Data Collection (IOEXEC) module provides for the routing of AOP commands and data to the various RTs, and for the collection of data that passes through the data bus between the AOP and the CDC-6600 simulation. The former takes packed data from the AOP, determines the target RT, unpacks the data and inserts the result in the RT's input buffer. The latter removes the data in the RT's input and output buffers to the AOP and inserts the unpacked data into the data collection buffer which will be transferred to a historical data base for data reduction.

The routine interrogates the input buffer from the AOP to determine the target RT and command, and transfers the associated unpacked data to the data collection buffer. The command word is then used to determine the type of information to be transferred to the RT. The data is then unpacked and transferred to the RT's input buffer. If the command is a data transfer from the RT to the AOP, the data in the RT's output buffer is to the AOP, that data is transferred to the data collection buffer.



Inputs and Outputs Between the AOP and the RT Modules
Figure G-1


IOEXEC
DO UNTIL ALL AOP COMMANDS ARE PROCESSED
DECODE COMMAND WORD
TRANSFER COMMAND AND DATA TO DATA COLLECTION BUFFER
COMMAND
MODE/ DISCRETE DATA
NORMAL DATA TRANSFER
TRANSMIT BIT STATUS
RETRANSMIT BIT STATUS
TRANSMIT LAST COMMAND
CONTROL COMMAND DATA TRANSFER
MULTI-MESSAGE TRANSFER
UNPACK & TRANSFER INIT, TERM, INIT. PROC. OR INIT. SELF-TEST INTO RT INPUT BUFFER
T
RT TO AOP TRANSFER
F
NO-OP COMMAND RESET RT'S "STATUS SENT" FLAG
NO-OP COMMAND RESET RT's "STATUS SENT" FLAG
NO-OP COMMAND RESET RT's "STATUS SENT" FLAG
UNPACK DATA WRDS & TRANSFER CONTROL COMMAND DATA INTO RT'S INPUT BUFFER
UNPACK DATA WRDS & TRANSFER INTO RT'S MULTI-MESSAGE INPUT BUFFER
TRANSFER DATA IN RT'S OUTPUT BUFFER TO DATA COLLECTION BUFFER
UNPACK DATA WRDS & TRANSFER NORMAL DATA INTO RT'S INPUT BUFFER
END DO UNTIL
RETURN

APPENDIX H

PROGRAM PROFILE

CONVERTER MULTIPLEXER MODULE

(CMUX)

```
      PROGRAM MUXDRIV
      ABSTRACT
           THIS PROGRAM IS A DRIVER TO TEST THE CMUX MODULE.

      CODINF HISTORY
           1. PROGRAMMED--ALEX PODLECKI (CSC) 01/12/78

      END OF ABSTRACT
```

```
INITIALIZE BUFFERS, FLAGS, ETC.
ZERO THE MUX INPUT BUFFER
DOWHILE INPUT BUFFER NOT INITIALIZED
.  . INSERT ZERO VALUE
ENDDO
ZERO THE ATO,SO DISPLAY BUFFERS
DOWHILE DISPLAY BUFFER NOT INITIALIZED
.  . INSERT ZERO VALUE
ENDDO
ZERO THE MUX ALTERNATE INPUT BUFFER
DOWHILE ALTERNATE BUFFER NOT INITALIZED
.  . INSERT ZERO VALUE
ENDDO
ZERO THE DATA TRANSFER HOLDING BUFFER
DOUNTIL HOLDING BUFFER EXHAUSTED
.  . INSERT ZERO WORD
ENDDO
NO INITIAL FAULTS
ZERO MAD, ACOUSTICS DISPLAY WORDS
ALTITUDE = 512 FEET
DUMMY UP PITCH/ROLL SINE/COSINE
GAMMAS = 511
TACAN RANGE = 5.13 NMI
INDICATED AIRSPEED = 37.5 KNOTS
VALID BEARING = 225 DEGREES
ATO STICK VOLTAGES - X=4.992, Y=2.496
SO STICK VOLTAGES - X=1.248, Y=0.624
ZERO DATA AVAILABLE WORDS
ALLOW PRINTING INFORMATIVE MESSAGES
RESPONSE TO NO COMMANDS
SELF-TEST SEQUENCE
1. INITIATE SELF-TEST MODE/DISCRETE
2. IDLE PERIOD
DOWHILE CMUX SAYS ITS BUSY
.  . EXECUTE CMUX
ENDDO
3. TRANSMIT *BIT* STATUS COMMAND
INITIALIZATION SEQUENCE
-INITIALIZE TERMINAL MODE/DISCRETE
-TRANSMIT BIT STATUS COMMAND
-INITIATE PROCESSING MODE/DISCRETE
REQUEST FOR DATA
1. NULL INPUT
2. NORMAL DATA TRANSFER WITH T/R = 1
3. NULL INPUT
DATA TRANSFER < 32 WORDS
1. DATA TRANSFER COMMAND WITH COUNT = 15
.  HEADER WORD 1 - IPL WITH COUNT = 15
.  HEADER WORD 2 - ADDRESS = 1
DOWHILE IPL WORDS REQUIRED
.  . INSERT DUMMY IPL WORD = POSITION IN BUFFER
ENDDO
2. NULL INPUT 4 TIMES
DATA TRANSFER > 32 WORDS
1. MULTI-MESSAGE COMMAND
.  HEADER WORD 1 - DISPLAY DATA WITH COUNT = 42
```

```
.  HEADER WORD 2 - ACOUSTICS DATA WITH COUNT = 1
.  HEADER WORD 3 - LOFAR, DIFAR ALI DATA IN ZONE 1
.  HEADER WORD 2 - ACOUSTICS DATA WITH COUNT = 1
.  HEADER WORD 3 - DEMON ALI DATA IN ZONE 2
.  HEADER WORD 2 - ACOUSTICS DATA WITH COUNT = 1
.  HEADER WORD 3 - RANGE DOPPLER DATA IN ZONE 4
.  HEADER WORD 2 - MAD DATA WITH COUNT = 1
.  HEADER WORD 3 - FULL SCALE
.  HEADER WORD 2 - ATO DISPLAY DATA WITH COUNT = 18
.  HEADER WORD 3 - ADDRESS = 2001B
DOWHILE ATO DATA REQUIRED
.  . INSERT DUMMY DATA = POSITION WITHIN BLOCK
ENDDO
.  HEADER WORD 2 - SO DISPLAY DATA WITH COUNT = 13
.  HEADER WORD 3 - ADDRESS = 3001B
.  DUMMY SO DATA TO PAD OUT MULTI-MESSAGE BLOCK
2. TRANSMIT STATUS WORD
3. NORMAL DATA TRANSFER OF 10 WORDS
DOWHILE SO DATA REQUIRED
.  . INSERT DUMMY DATA = POSITION IN TRANSFER
ENDDO
```

```
      SUBROUTINE XCMUX
      ABSTRACT
           THIS ROUTINE MONITORS EXECUTION OF THE CMUX MODULES
           FOR THE CMUX DRIVER.
           1. IT PRINTS CONTENTS OF CURRENT INPUT BUFFER
           2. IT PRINTS CHANGES WITHIN ALTERNATE INPUT BUFFER
           3. IT EXECUTES THE CMUX1 MODULE
           4. IT PRINTS CHANGES TO THE DATA TRANSFER HOLDING BUFFER
           5. EVERY 5TH CYCLE, IT EXECUTES CMUX2
           6. IT PRINTS THE CONTENTS OF THE RESULTANT OUTPUT BUFFER
              AND RESETS THE PP BIT TO SIGNIFY ACCEPTANCE
           7. IT PRINTS CHANGES TO THE ATO/SO BUFFERS
           8. IT PRINTS MAD, ACOUSTICS FLAGS/DATA

      CODING HISTORY
           1. PROGRAMMED--ALEX PODLECKI (CSC) 01/16/78

      END OF ABSTRACT
```

```
PRINT INPUT BUFFER
DOWHILE SOMETHING IN INPUT BUFFER
.  . EXPAND INPUT WORD
.  . PRINT INPUT WORD BIT-BY-BIT
ENDDO
IF EMPTY INPUT BUFFER AND MESSAGES REQUESTED
. THEN
.  . PRINT INFORMATIVE MESSAGE
. ELSE
.  . OMIT MESSAGE
ENDIF
DOUNTIL ALTERNATE BUFFER EXHAUSTED
.  . IF THE ALTERNATE BUFFER HAS CHANGED
.  .  .THEN
.  .  .  .IF THIS IS THE INITIAL CHANGE
.  .  .  .  THEN
.  .  .  .  .  PRINT HEADINGS FOR NEW VALUES
.  .  .  .  ELSE
.  .  .  .  .  OMIT HEADING MESSAGE
.  .  .  .ENDIF
.  .  .  .PRINT POSITION AND NEW VALUE
.  .  .  .SAVE NEW VALUE
.  .  .ELSE
.  .  .  .OMIT PRINTING
.  . ENDIF
ENDDO
SET UP DUMMY HEADING SIN/COS
EXECUTE THE CMUX1 MODULE
DOUNTIL HOLDING BUFFER EXHAUSTED
.  . IF THE HOLDING BUFFER HAS CHANGED
.  .  .THEN
.  .  .  .IF THIS IS THE INITIAL CHANGE
.  .  .  .  THEN
.  .  .  .  .  PRINT HEADINGS FOR NEW VALUES
.  .  .  .  ELSE
.  .  .  .  .  OMIT MESSAGE
.  .  .  .ENDIF
.  .  .  .PRINT POSITION AND NEW VALUE
.  .  .  .SAVE NEW VALUE
.  .  .ELSE
.  .  .  .OMIT PRINTING
.  . ENDIF
ENDDO
IF 5TH CYCLE
. THEN
.  . EXECUTE THE CMUX2 MODULE
.  . IF SOMETHING IN OUTPUT BUFFER
.  .  .THEN
.  .  .  .PRINT HEADER WORD
.  .  .  .DOWHILE SOMETHING IN OUTPUT BUFFER
.  .  .  .  .  EXPAND OUTPUT WORD
.  .  .  .  .  IF THIS WORD IS A BIT STATUS WORD
.  .  .  .  .  . THEN
.  .  .  .  .  .  . INDICATE BY *
.  .  .  .  .  . ELSE
.  .  .  .  .  .  . INDICATE BY BLANK
```

```
. . . . . ENDIF
. . . . . PRINT OUTPUT WORD BIT-BY-BBIT
. . . .ENDDO
. . . .TOGGLE THE PP DATA AVAILABLE BIT
. . .ELSE
. . . .IF MESSAGES REQUESTED
. . . . THEN
. . . . . PRINT INFORMATIVE MESSAGE
. . . . ELSE
. . . . . OMIT INFORMATIVE MESSAGE
. . . .ENDIF
. . ENDIF
. . DOUNTIL ATO/SO BUFFERS EXHAUSTED
. . . .IF EITHER ATO OR SO BUFFER HAS CHANGED
. . . . THEN
. . . . . IF THIS IS THE INITIAL CHANGE
. . . . . THEN
. . . . . PRINT HEADINGS
. . . . . ELSE
. . . . . OMIT HEADINGS
. . . . . ENDIF
. . . . . PRINT POSITION AND CURRENT VALUES
. . . . . SAVE CURRENT VALUES
. . . . ELSE
. . . . . OMIT PRINTING
. . . .ENDIF
. . ENDDO
. . IF MAD DATA AVAILABLE
. . . .THEN
. . . . .PRINT MAD SCALE FACTOR, DISPLAY ZONE
. . . .ELSE
. . . . .OMIT PRINTING
. . ENDIF
. . DOFOR EACH DISPLAY ZONE
. . . . .IF ACOUSTICS DATA PRESENT
. . . . . THEN
. . . . . PRINT ZONE AND TYPE
. . . . . ELSE
. . . . . OMIT PRINTING
. . . . .ENDIF
. . ENDDO
. ELSE
. . DO NOT EXECUTE THE CMUX2 MODULE
ENDIF
```

```
      SUBROUTINE ADVANCE(POINTER,LWA)
      ABSTRACT
           THIS ROUTINE INCREMENTS A POINTER BY 1. IF THE POINTER WAS
           ALREADY SET TO AN LWA, THE POINTER IS RESET TO 1.

           POINTER - CURRENT VALUE OF POINTER

           LWA - LAST WORD ADDRESS FOR POINTER

      CODING HISTORY
           1. PROGRAMMED--ALEX PODLECKI        11/04/77

      END OF ABSTRACT
```

```
IF POINTER IS LESS THAN LWA
. THEN
.   . INCREMENT POINTER
. ELSE
.   . SET POINTER TO FWA
ENDIF
```

```
      SUBROUTINE EXPAND(N,IN,OUT)
      ABSTRACT
           THIS ROUTINE EXPANDS A WORD INTO AN N WORD ARRAY SUCH THAT
           WORD 1 CONTAINS BIT N-1, WORD 2 CONTAINS BIT N-2, ...,
           AND WORD N CONTAINS BIT 0 ( RIGHT JUSTIFIED WITH ZERO FILL )

           N - NUMBER OF BITS TO BE EXPANDED

           IN - INPUT WORD TO BE EXPANDED

           OUT - OUTPUT ARRAY TO RECEIVE EXPANSION

      CODING HISTORY
           1. PROGRAMMED--ALEX PODLECKI        11/07/77

      END OF ABSTRACT
```

```
DOWHILE ANOTHER BIT TO BE EXPANDED
.  . MASK OUT DESIRED BIT
.  . SET UP FOR NEXT BIT
ENDDO
```

```
      SUBROUTINE PACKPP( NRT, N)
      ABSTRACT
           THIS ROUTINE IS A DUMMY SUBSTITUTE FOR THE ACTUAL PACKPP.

           NRT - NUMBER CORRESPONDING TO RT

           N - NUMBER OF WORDS TO BE *PACKED*

      CODING HISTORY
           1. PROGRAMMED--ALEX PODLECKI        11/07/77

      END OF ABSTRACT
```

EXIT

```
      SUBROUTINE CMUX1
      ABSTRACT
           THIS PROGRAM PROCESSES ALL AYK COMMANDS TO THE CMUX,
           BUILDS A HOLDING BUFFER FOR ALL DATA TRANSFERS,
           CALLS CMUXCDT TO UPDATE DISPLAY BUFFERS, TABLES AND FLAGS
           AND SAVES 4 SETS OF HELO HEADING SINE AND COSINE.

      CODING HISTORY
           1. PROGRAMMED--ALEX PODLECKI  12/28/77

      END OF ABSTRACT
```

```
MAIN LOOP FOR INPUT PROCESSING
IF SOMETHING IN THE INPUT BUFFER
. THEN
.  . RESET STATUS SENT FLAG
. ELSE
ENDIF
DOWHILE SOMETHING IN INPUT BUFFER
.  . CRACK COMMAND INTO BASIC FIELDS
.  . CASE OF AOP COMMAND WORD (COMMAND)
.  .  .  .*COMMAND EQ. 0
.  .  .  .CASE OF MODE/DISCRETE COMMAND (DATAWC)
.  .  .  .  .  *DATAWC EQ. 1
.  .  .  .  .  INITIALIZE TERMINAL
.  .  .  .  .  RESET SELF-TEST COUNTER/FLAG
.  .  .  .  .  *DATAWC EQ. 3
.  .  .  .  .  INITIALIZE SELF-TEST
.  .  .  .  .  SET SELF-TEST COUNTER TO MAXIMUM
.  .  .  .  .  *DATAWC EQ. 4
.  .  .  .  .  INITIATE PROCESSING
.  .  .  .  .  SET PROCESSING INITIATED AND DATA REQUESTED FLAGS
.  .  .  .END CASE
.  .  .  .*COMMAND EQ. 6
.  .  .  .MULTI-MESSAGE TRANSFER
.  .  .  .DOWHILE ANOTHER WORD IN ALTERNATE CMUX BUFFER
.  .  .  .  .  TRANSFER WORD TO MULTI-MESSAGE HOLD BUFFER
.  .  .  .ENDDO
.  .  .ELSE
.  .  .  .*COMMAND EQ. 1
.  .  .  .NORMAL DATA TRANSFER
.  .  .  .IF A4K IS REQUESTING DATA
.  .  .  .  THEN
.  .  .  .  .  SET DATA REQUESTED FLAG
.  .  .  .  ELSE
.  .  .  .  .  IF DATA WORD COUNT FIELD IS ALL ZEROS
.  .  .  .  .  . THEN
.  .  .  .  .  .  . CHANGE WORD COUNT TO 32
.  .  .  .  .  . ELSE
.  .  .  .  .  .  . LEAVE DATA WORD COUNT AS IS
.  .  .  .  .  ENDIF
.  .  .  .  .  DOWHILE ANOTHER DATA WORD IS AVAILABLE
.  .  .  .  .  .  . TRANSFER INPUT WORD TO HOLD BUFFER
.  .  .  .  .  .  . ZERO INPUT BUFFER
.  .  .  .  .  ENDDO
.  .  .  .  .  PROCESS THE COMPLETED DATA TRANSFER
.  .  .  .  .  ZERO THE HOLDING BUFFER FOR COMPLETE DATA TRANSFERS
.  .  .  .ENDIF
.  . END CASE
.  . ENDIF
ENDDO
SAVE 4 SETS OF HEADING SIN/COS
END OF PROGRAM
```

```
      SUBROUTINE CMUXCDT
      ABSTRACT
           THIS ROUTINE PROCESSES COMPLETED DATA TRANSFERS FROM THE AYK
           TO THE CMUX. MAD AND ACOUSTICS FLAGS AND ATO/SO DISPLAY
           BUFFERS ARE UPDATED.

      CODING HISTORY
           1. PROGRAMMED--ALEX PODLECKI (CSC) 12/30/77

      END OF ABSTRACT
```

```
IF PROCESSING NOT INITIATED OR SELF-TEST IN PROGRESS
. THEN
.  . SET EXIT FLAG
. ELSE
.  . CONTINUE PROCESSING
ENDIF
EXTRACT BUFFER COUNT FROM HEADER WORD 1
IF BUFFER COUNT IN ERROR
. THEN
.  . SET BUFFER COUNT ERROR FLAG
.  . SET EXIT FLAG
. ELSE
.  . CONTINUE PROCESSING HEADER WORD 1
ENDIF
IF IPL DATA IS BEING RECEIVED
. THEN
.  . IF UNREASONABLE BUFFER ADDRESS
.  .  .THEN
.  .  .  .SET HEADER WORD ERROR FLAG
.  .  .ELSE
.  .  .  .SET IPL OCCURRED FLAG
.  . ENDIF
.  . SET EXIT FLAG
. ELSE
.  . CONTINUE ONTO NEXT HEADER WORD
ENDIF
IF EXIT FLAG NOT SET
. THEN
.  . DOWHILE DATA BLOCK IS IN BUFFER AND NO ERRORS
.  .  .  .HEADER WORD 2 PROCESSING
.  .  .  .EXTRACT BLOCK COUNT
.  .  .  .IF THIS NEXT BLOCK IS WITHIN THE HOLDING BUFFER
.  .  .  .  THEN
.  .  .  .  .  CASE OF DISLAY DATA (ITYPE)
.  .  .  .  .  .  .  . *ITYPE EQ. 8
.  .  .  .  .  .  .  . DISPLAY DATA BIT CODE IS 1000
.  .  .  .  .  .  .  . ACOUSTICS DISPLAY DATA
.  .  .  .  .  .  .  . SAVE ACOUSTICS DATA TYPE AND DISPLAY ZONE
.  .  .  .  .  .  .  . CONTENTS
.  .  .  .  .  .  .  . HEADER WORD 3 BREAKDOWN
.  .  .  .  .  .  . ELSE
.  .  .  .  .  .  .  . *ITYPE EQ. 1
.  .  .  .  .  .  .  . DISPLAY DATA BIT CODE IS 0001
.  .  .  .  .  .  .  . MAD DISPLAY DATA
.  .  .  .  .  .  .  . SET MAD DATA FLAG
.  .  .  .  .  .  .  . SAVE RE-SCALE FACTOR
.  .  .  .  .  .  .  . *ITYPE EQ. 4
.  .  .  .  .  .  .  . DISPLAY DATA BIT CODE 0100
.  .  .  .  .  .  .  . ATO DISPLAY DATA
.  .  .  .  .  .  .  . EXTRACT ATO BUFFER ADDRESS
.  .  .  .  .  .  .  . IF UNREASONABLE ADDRESS
.  .  .  .  .  .  .  .  .THEN
.  .  .  .  .  .  .  .  .  .SET HEADER WORD ERROR FLAG
.  .  .  .  .  .  .  .  .  .SET EXIT FLAG
.  .  .  .  .  .  .  .  .ELSE
.  .  .  .  .  .  .  .  .  .DOWHILE ATO DATA REMAINS IN THIS BLOCK
```

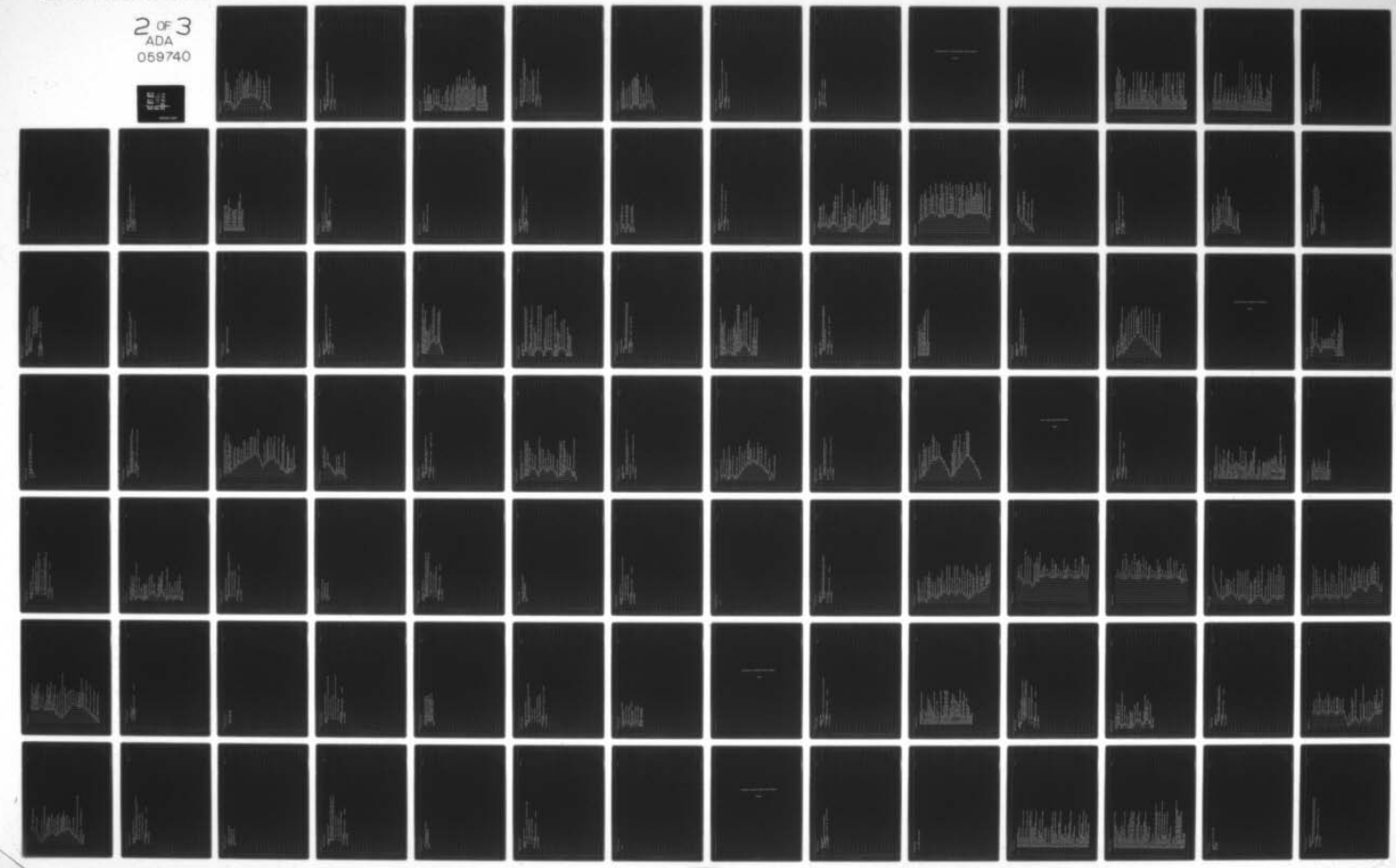
```
. . . . . . . . . . . . TRANSFER DATA TO ATC DISPLAY BUFFER
. . . . . . . . . . .ENDDO
. . . . . . . . . ENDIF
. . . . . . . . . *ITYPE EQ. 2
. . . . . . . . . DISPLAY DATA BIT CODE IS 0010
. . . . . . . . . SO DISPLAY DATA
. . . . . . . . . EXTRACT SO BUFFER ADDRESS
. . . . . . . . . IF UNREASONABLE ADDRESS
. . . . . . . . . .THEN
. . . . . . . . . . .SET HEADER WORD ERROR FLAG
. . . . . . . . . . .SET EXIT FLAG
. . . . . . . . . .ELSE
. . . . . . . . . . .DOWHILE SO DATA REMAINS IN THIS BLOCK
. . . . . . . . . . . . TRANSFER DATA TO SO DISPLAY BUFFER
. . . . . . . . . . .ENDDO
. . . . . . . . . ENDIF
. . . . . . . . . *ITYPE NE. 1,2,4 OR 6
. . . . . . . . . SET HEADER WORD ERROR FLAG
. . . . . . . . . SET EXIT FLAG
. . . . . . . . END OF CASE
. . . . . ELSE
. . . . . SET HEADER WORD ERROR FLAG
. . . . . SET EXIT FLAG
. . . . .ENDIF
. . . . .ADVANCE HOLDING BUFFER POINTER TO NEXT POSSIBLE HEADER
. . . . . WORD 2
. . . ENDDO
. . ELSE
. . . IGNORE DATA IN BUFFER
. . ENDIF
```

```
      SUBROUTINE CMUX2
      ABSTRACT
           THIS ROUTINE PERFORMS CMUX OUTPUT PROCESSING

      CODING HISTORY
           1. PROGRAMMED--ALEX PODLECKI (CSC) 01/04/78

      END OF ABSTRACT
```

```
IF SELF-TEST IS IN PROGRESS
. THEN
.  . DECREMENT SELF-TEST COUNTER
.  . IF SELF-TEST HAS NOT TIMED OUT
.  .  .THEN
.  .  .  .SELF-TEST TIMED OUT, SETUP OUTPUT OF
.  .  .  .RT AND BIT STATUS WORDS
.  .  .ELSE
.  .  .  .SELF-TEST NOT TIMED OUT
.  . ENDIF
. ELSE
.  . CONTINUE PROCESSING
.  . IF BUFFER COUNT ERROR HAS OCCURRED
.  .  .THEN
.  .  .  .SET BIT 12 OF FIRST BIT STATUS WORD
.  .  .ELSE
.  .  .  .LEAVE BIT STATUS WORD ALONE
.  . ENDIF
.  . IF HEADER WORD ERROR
.  .  .THEN
.  .  .  .SET BIT 12 IN SECOND BIT STATUS WORD
.  .  .ELSE
.  .  .  .LEAVE BIT STATUS WORD ALONE
.  . ENDIF
.  . IF POWER OFF/ON TRANSIENT HAS OCCURED
.  .  .THEN
.  .  .  .SET FLAGS TO PROCESSING HALTED
.  .  .ELSE
.  . ENDIF
.  . IF BIT STATUS HAS CHANGED
.  .  .THEN
.  .  .  .PLACE NEW FAULTS INTO OUTPUT BUFFER
.  .  .ELSE
.  .  .  .OMIT BIT STATUS WORDS FROM OUTPUT
.  . ENDIF
.  . SET UP RT STATUS WORD
.  . IF ANY OF THE MESSAGE ERROR BITS ARE ON
.  .  .THEN
.  .  .  .SET BIT 10 OF RT STATUS WORD
.  .  .ELSE
.  .  .  .LEAVE RT STATUS WORD ALONE
.  . ENDIF
.  . IF ANY OF THE *BIT* TEMP-HIGH BITS ARE ON
.  .  .THEN
.  .  .  .SET BIT 6 OF THE RT STATUS WORD
.  .  .ELSE
.  .  .  .LEAVE RT STATUS WORD ALONE
.  . ENDIF
.  . IF ANY OF THE OTHER ERROR BITS ARE ON
.  .  .THEN
.  .  .  .SET BIT 0 OF RT STATUS WORD
.  .  .ELSE
.  .  .  .LEAVE RT STATUS WORD ALONE
.  . ENDIF
.  . IF DATA REQUESTED, PROCESSING STARTED, IPL OCCURRED
.  . AND NOT SELF-TESTING
```

AD-A059 740 COMPUTER SCIENCES CORP HUNTINGDON VALLEY PA F/G 15/1
LAMPS SEAS SIMULATION SOFTWARE SUPPORT.(U)
JUN 78 N62269-75-C-0001
UNCLASSIFIED NL

2 OF 3
ADA
059740



```
. .  .THEN
. .  .  .CONSTRUCT CMUX OUTPUT NORMAL DATA TRANSFER
. .  .  .SET WORD COUNT IN RT STATUS
. .  .  .SET RT DATA AVAILABLE BIT
. .  .ELSE
. .  .  .OMIT OUTPUT DATA
. . ENDIF
ENDIF
IF PREVIOUS OUTPUT READ BY PP
. THEN
. . IF AOP HAS TAKEN THE PREVIOUS BUFFER
. .  .THEN
. .  .  .IF THERE IS NEW DATA TO BE SENT TO AOP
. .  .  .  THEN
. .  .  .  .  PUT RT STATUS WORD INTO OUTPUT BUFFER
. .  .  .  .  PACK OUTPUT BUFFER
. .  .  .  .  CONSTRUCT HEADER WORD
. .  .  .  .  SAVE CURRENT 'BIT' STATUS'
. .  .  .  .  RESET CP DATA AVAILABLE BIT
. .  .  .  .  IF DATA IS BEING TRANSMITTED TO AOP
. .  .  .  .  . THEN
. .  .  .  .  .  . SET DATA SENT STATUS FLAG
. .  .  .  .  . ELSE
. .  .  .  .  .  . FLAG NOT SET
. .  .  .  .  ENDIF
. .  .  .  .  SET FLAG THAT THIS DATA HAS NOT BEEN
. .  .  .  .  ACKNOWLEDGED / REQUESTED YET
. .  .  .  ELSE
. .  .  .  .  LEAVE OUTPUT BUFFER EMPTY
ENDIF
. .  .ELSE
. .  .  .AOP HAS NOT TAKEN LAST BUFFER
. .  .  .SET ERROR WORDS 2 AND 3
. . ENDIF
. ELSE
. . PP DID NOT TAKE BUFFER, SET ERROR WORD 1
ENDIF
```

```
      SUBROUTINE CMXDATA
      ABSTRACT
           THIS ROUTINE CONSTRUCTS THE 31 WORD DATA ARRAY USED AS THE
           CMUX NORMAL DATA TRANSFER TO THE AYK

      CODING HISTORY
           1. PROGRAMMED--ALEX PODLECKI (CSC) 01/10/78

      END OF ABSTRACT
```

```
      WORD 1 - ELECTRONIC ALTIMETER
      USE HELO(15)
      WORD 2 - TACAN RANGE
      PACK TACAN RANGE WITH MSB = 327.68
      WORD 3 - TACAN BEARING
      IF VALID BEARING DATA
      . THEN
      .  . IF LESS THAN 180 DEGREES
      .  .  .THEN
      .  .  .  .PACK BEARING AS IS
      .  .  .ELSE
      .  .  .  .PACK BEARING - 180
      .  . ENDIF
      . ELSE
      .  . SEND 'NO DATA'
      ENDIF
      WORD 4,5 - ATO STICK X,Y
      DOFOR ATO X,Y STICK VOLTAGES
      .  . PACK VOLTAGE WITH LSB=0.0195
      ENDDO
      WORD 6,7 - SO STICK X,Y
      DOFOR SO X,Y STICK VOLTAGES
      .  . PACK VOLTAGES WITH LSB=0.0195
      ENDDO
      WORD 8,9 - PITCH SINE,COSINE
      CONVERT PITCH SINE TO 2'S COMPLEMENT BINARY
      STORE INTO CMUX TEMPORARY BUFFER
      CONVERT PITCH COSINE TO 2'S COMPLEMENT BINARY
      STORE INTO CMUX TEMPORARY BUFFER
      WORD 10,11 - ROLL SINE,COSINE
      CONVERT ROLL SINE TO 2'S COMPLEMENT BINARY
      STORE INTO CMUX TEMPORARY BUFFER
      CONVERT ROLL COSINE TO 2'S COMPLEMENT BINARY
      STORE INTO CMUX TEMPORARY BUFFER
      WORD 12-27 - HEADING 4 SAMPLES, 2 SOURCES, SINE,COSINE
      DOUNTIL 4 SAMPLES GENERATED
      .  . CONVERT SINE TO 2'S COMPLEMENT BINARY
      .  . STORE AS BOTH HEADING SINES
      .  . CONVERT COSINE TO 2'S COMPLEMENT BINARY
      .  . STORE AS BOTH HEADING COSINES
      ENDDO
      WORD 28 - INDICATED AIRSPEED
      PACK HELO(21) CONVERTED TO KNOTS
      WORD 29 - BAROMETRIC ALTITUDE
      PACK HELO(15)
      WORD 30 - OUTSIDE AIR TEMPERATURE
      IT'S ALWAYS 25 DEGREES CENTIGRADE
      WORD 31 - LATEST MAD CONVERSION
      PACK GAMMAS
```

```
      FUNCTION MUXPACK( SOURCE, N, SIGBIT)
      ABSTRACT
           THIS FUNCTION PACKS A REAL INTO A BINARY OF REQUESTED SIZE
           THE RESULT HAS THE SIGN BIT EXTENDED THROUGH 60 BITS

           SOURCE = REAL VALUE TO BE PACKED

           N = NUMBER OF BITS IN RESULT

           SIGBIT = REAL VALUE OF MSB OR LSB IN RESULT
           ( MSB IF POSITIVE, LSB IF NEGATIVE)

      CODING HISTORY
           1. PROGRAMMED--ALEX PODLECKI (CSC) 01/27/78

      END OF ABSTRACT
```

```
INITIALIZE RESULT VALUE
WORK ONLY WITH POSITIVE SOURCE VALUES
IF LSB VALUE WAS SUPPLIED
. THEN
.  . CALCULATE THE MSB VALUE
. ELSE
.  . USE THE MSB VALUE SUPPLIED
ENDIF
DOUNTIL ALL BITS DETERMINED IN RESULT
.  . SHIFT PREVIOUS RESULT OVER ONE BIT
.  . DETERMINE VALUE OF NEXT BIT IN RESULT
.  . MERGE IN NEXT BIT WITH PREVIOUS BITS
.  . CALCULATE NEW REMAINDER FROM SOURCE VALUE
.  . CALCULATE REAL VALUE OF NEXT BIT
ENDDO
IF INITIAL VALUE WAS NEGATIVE
. THEN
.  . COMPLEMENT RESULT
. ELSE
.  . IF RESULT HAS SPILLED OVER INTO SIGN BIT
.  .  .THEN
.  .  .  .SET RESULT TO *N* ONES
.  .  .ELSE
.  .  .  .LEAVE RESULT AS IS
.  . ENDIF
ENDIF
```

```
      FUNCTION K2SCOMP(NUMBER)
      ABSTRACT
           OBTAINS THE 2'S COMPLEMENT FORM OF A 1'S COMPLEMENT NUMBER

      CODING HISTORY
           1. PROGRAMMED--ALEX PODLECKI (CSC) 01/30/78

      END OF ABSTRACT
```

```
IF NUMBER IS NEGATIVE
. THEN
.   . 2'S COMPLEMENT = 1'S COMPLEMENT + 1
. ELSE
.   . 2'S COMPLEMENT = 1'S COMPLEMENT
ENDIF
```

COMMUNICATION SYSTEM CONTROL GROUP MODULE

(CSCG)

```
      PROGRAM DRIVER
      ABSTRACT
           DRIVER PROGRAM FOR CSCG SOFTWARE MODULE.

      CODING HISTORY
           1.  PROGRAMMED  J. MANGES   CSC  MARCH, 1978

      END OF ABSTRACT
```

```
SET INITIAL CONDITIONS
MAKE SURE UHF AUTO IS ON  (SOFTWARE CONTROL)
INITIALIZE DATA WORDS TO INDICATE UHF-1 IS IN OTPI MODE
SET DATA WORDS TO INDICATE DATA LINK IS IN ASW MODE
SET ALL RECEIVER SIGNAL STRENGTHS TO MAXIMUM
ENDDO
ASSIGN THE BUOY IN CHUTE 1 TO SONO RCVR A
ASSIGN THE BUOY IN CHUTE 2 TO SONO RCVR B
AND ETC.
ENDDO
CHECK INITIALIZATION
CHECK ACTION OF ERROR WORDS
ERROR WORD 2
SET STATUS SENT FLAG TO 1
CHECK VARIABLES, CALL CSCG, AND CHECK THE RESULTS
ERROR WORD 3
SENT DATA SENT FLAG TO 1
CHECK VARIABLES, CALL CSCG, AND CHECK THE RESULTS
ERROR WORD 1
SET DATA AVAILABLE FLAGS TO OPPOSITE VALUES
CHECK VARIABLES, CALL CSCG, AND CHECK THE RESULTS
RESET ERROR WORDS AND FLAGS
CHANGE IN BIT STATUS WORD
SET A BIT IN THE BIT STATUS WORD
CHECK VARIABLES, CALL CSCG, AND CHECK THE RESULTS
RESET FLAGS
CSCG SELF TEST SEQUENCE
SET COMMAND WORD INTO THE INPUT BUFFER
SET BUFFER POINTER
CHECK VARIABLES, CALL CSCG, AND CHECK THE RESULTS
RESET FLAGS
DECREMENT THE BIT COUNTER
DO WHILE I IS LESS THAN 147
ENDDO
DO WHILE I IS LESS THAN THREE
ENDDO
CSCG INITIALIZATION SEQUENCE
SET INITIALIZE TERMINAL COMMAND IN INPUT BUFFER
SET BUFFER POINTER
CHECK VARIABLES, CALL CSCG, AND CHECK THE RESULTS
SET INITIATE PROCESSING COMMAND IN INPUT BUFFER
SET BUFFER POINTER
CHECK VARIABLES, CALL CSCG, AND CHECK THE RESULTS
RESET FLAGS
CSCG NORMAL DATA TRANSFER SEQUENCE
SET NORMAL DATA TRANSFER COMMAND IN INPUT BUFFER
SET BUFFER POINTER
SET DATA SENT FLAG TO 1
CHECK VARIABLES, CALL OLCS, AND CHECK THE RESULTS
RESET FLAGS
SET NORMAL DATA TRANSFER COMMAND IN INPUT BUFFER
SET TEST DATA WORD BLOCK IN INPUT BUFFER
SET BUFFER POINTER
CHECK VARIABLES, CALL CSCG, AND CHECK THE RESULTS
RESET FLAGS
CSCG DISCRETES
```

```
CHANGE UHF MODE
CHANGE SWITCH SETTING FROM OTPI TO ADF MODE
CHECK VARIABLES, CALL CSCG, AND CHECK THE RESULTS
RESET FLAGS
CHANGE UHF MODE
CHANGE SWITCH SETTING FROM ADF MODE TO OTPI MODE
CHECK VARIABLES, CALL CSCG, AND CHECK THE RESULTS
RESET FLAGS
SET UHF CHANNEL TO 1
SET UHF-1 CHANNEL NUMBER TO 1
RESET FLAGS
SET UHF CHANNEL TO 6
SET UHF-1 CHANNEL NUMBER TO 6
RESET FLAGS
SET UHF CHANNEL TO 18
SET UHF-1 CHANNEL NUMBER TO 18
RESET FLAGS
SET UHF CHANNEL TO 32
SET UHF-1 CHANNEL NUMBER TO 32
RESET FLAGS
-------------------------------------------------------------
CSCG EXTERNAL INPUTS
-------------------------------------------------------------
D/L MODE
SET D/L MODE TO ASMD
CHECK VARIABLES, CALL CSCG, AND CHECK THE RESULTS
RESET FLAGS
SONOBUOY RECEIVER SIGNAL STRENGTH
SET THE POSTION OF THE HELO
SET THE POSITION OF THE BUOYS
CHECK VARIABLES, CALL CSCG, AND CHECK THE RESULTS
RESET FLAGS
SHIFT HELO POSITION
CHECK VARIABLES, CALL CSCG, AND CHECK THE RESULTS
RESET FLAGS
SHIFT HELO ALTITUDE
CHECK VARIABLES, CALL CSCG, AND CHECK THE RESULTS
OTPI BEARING FOR CASS BUOY IN CHUTE 2
INDICATE CASS BUOY IN CHUTE 2
SET VHF TRANS. ON
SET PING TIME
SET IN WATER FLAG
SET RF NUMBER FOR BUOY IN CHUTE 2
SET RF EQUAL TO 10 IN KEYSET
MAKE SURE UHF IN OTPI MODE
CHECK VARIABLES, CALL CSCG, AND PRINT THE RESULTS
```

```
SUBROUTINE FLAGS
ABSTRACT
     THIS ROUTINE RESETS THE VALUES OF THE BUFFER AND DATA
     FLAGS TO A STATE IN WHICH THE PACKPP ROUTINE MAY BE CALLED.

CODING HISTORY
     1.  PROGRAMMED-  J. MANGES   CSC  APRIL 11, 1978

END OF ABSTRACT
```

RESET OUTPUT BUFFER FULL FLAGS
EQUATE DATA AVAILABLE AND PP DATA AVAILABLE FLAGS

```
      SUBROUTINE PRINT
      ABSTRACT
           PRINT DISPLAYS THE VALUES OF ALL RELEVANT CSCG VARIABLES
           WHEN CALLED BY THE DRIVER PROGRAM.
      CODING HISTORY
           1. PROGRAMMED  J. MANGES  CSC  DEC 1977
      END OF ABSTRACT
```

```
      WRITE OUT THE VALUES OF THE FLAGS
      WRITE OUT BIT BY BIT THE RT WORDS
      WRITE OUT THE DATA WORDS
      WRITE OUT THE CONTENTS OF THE INPUT BUFFER
      DO WHILE I IS LESS THAN TWENTY
      ENDDO
      WRITE OUT THE DATA LINK MODE
      WRITE OUT THE OTPI BEARING
      WRITE OUT THE DISCRETE ARRAY
      DO WHILE I IS LESS THAN THREE
      ENDDO
      WRITE OUT THE VALUE OF THE BIT COUNTER
      WRITE OUT THE VALUES OF THE INPUT BUFFER POINTERS
      WRITE OUT THE ERROR FLAGS
      DO WHILE I IS LESS THAN THREE
      ENDDO
```

```
      SUBROUTINE BITS(JVALUE,NUM)
      ABSTRACT
           THIS SUBROUTINE PRINTS OUT BIT BY BIT THE FIRST 16 BITS
           OF THE WORD JVALUE.
      CODING HISTORY
           PROGRAMMED  1. J. MANGES  CSC  12/28/77
      END OF ABSTRACT
```

```
      DO WHILE I IS LESS THAN 16
      .  . ENDIF
      ENDDO
      WRITE OUT THE VALUES OF THE BIT ARRAY
```

```
      SUBROUTINE DATAWDS
      ABSTRACT
           THIS SUBROUTINE PRINTS OUT ALL THE CSCG DATA WORDS WHEN
           CALLED BY THE PRINT SUBROUTINE.
      CODING HISTORY
           1.  PROGRAMMED  J. MANGES  CSC  FEB, 1978
      END OF ABSTRACT
```

```
      DO WHILE J IS LESS THAN 17
      .  . DO WHILE I IS LESS THAN SIXTEEN
      .  . ENDDO
      ENDDO
      DO WHILE J IS LESS THAN TWENTY NINE
      .  . DO WHILE I IS LESS THAN SIXTEEN
      .  . ENDDO
      ENDDO
      DO WHILE J IS LESS THAN SEVENTEEN
      DO WHILE J IS LESS THAN TWENTY-NINE
      ENDDO
```

```
      SUBROUTINE CSCG
      ABSTRACT
           SOFTWARE SIMULATION OF THE DATA INFORMATION TRANSFER SET.

      CODING HISTORY
           1.  PROGRAMMED  J. MANGES  CSC  JAN-MARCH  1978

      END OF ABSTRACT
```

```
. . POINTER
. . CHECK FOR BUFFER WRAP AROUND
. . IF IB IS GREATER THAN NUMWDS
. .  .THEN
. .  .  .RESET POINTER TO BEGINNING
. .  .ELSE
. .  .  .CHECK FOR EOI
. . ENDIF
. . CHECK FOR EOI
. . IF IB EQUALS IF
. .  .THEN
. .  .  .RESET NWOUTRT(6)
. .  .  .BUFFER PROCESSING IS COMPLETED
. .  .  .JUMP TO BIT COUNTER PROCESSING SECTION
. .  .ELSE
. .  .  .CONTINUE TO READ IN FROM INPUT BUFFER
. . ENDIF
ELSE
. . NOTHING IN INPUT BUFFER SO SKIP INPUT PROCESSING THIS CALL
ENDIF
DO BIT COUNTER PROCESSING
IF BIT IS IN PROGRESS
. THEN
. . DECREASE THE BIT COUNTER BY ONE
. . CHECK FOR END OF BIT
. . IF NCOUNTR IS EQUAL TO ZERO
. .  .THEN
. .  .  .BIT HAS ENDED SO SET RECEIVE BUSY BIT TO ZERO
. .  .ELSE
. .  .  .BIT IS ON SO CONTINUE
. . ENDIF
. ELSE
. . CONTINUE ON- BIT NOT IN PROGRESS
ENDIF
CHECK FOR QUIESCENT STATE
IF CSCG NOT IN A QUEISCENT STATE
. THEN
. . CHECK FOR BIT SELF TEST IN PROGRESS
. . IF BIT IS NOT ON OR HAS ONLY JUST BEGUN
. .  .THEN
. .  .  .CHECK FOR MANUAL OR AUTO UHF MODE
. .  .  .IF UHF IS IN MANUAL MODE
. .  .  .  THEN
. .  .  .  .  READ IN OTPI/ADF MODE SETTING AND CHANNEL SELECTION
. .  .  .  .  FROM DISCRETES
. .  .  .  ELSE
. .  .  .  .  LEAVE UHF-1 SETTINGS AS RECEIVED VIA SOFTWARE
. .  .  .ENDIF
. .  .  .UPDATE STATUS OF CSCG PERIPHERALS AND MAKE DATA WORD CHA
. .  .  .(O/L MODE AND SONOBUOY RCVRS SIGNAL STRENGTH)
. .  .  .OUTPUT SONOBUOY RECEIVER UNIT CHANNEL SELECTIONS TO
. .  .  .SONOBOY ROUTINE
. .  .  .OUTPUT OTPI BEARING TO DTOA ROUTINE
. .  .  .CHECK TO SEE IF PP DATA AVBL FLAG AND DATA AVBL FLAG
. .  .  .ARE EQUAL
. .  .  .IF IPPDATA IS EQUAL TO IDATAVB
```

```
. . . . THEN
. . . . . CHECK OUTPUT BUFFER FULL FLAGS
. . . . . IF IBFUL1(6) AND IBFUL2(6) ARE ZERO
. . . . . . THEN
. . . . . . . INITIALIZE WORD COUNTER TO ONE
. . . . . . . CHECK FOR CHANGE IN BIT STATUS WORD
. . . . . . . IF IUPS NOT EQUAL TO ZERO
. . . . . . . . .THEN
. . . . . . . . . .LOAD THE INPUT ARRAY
. . . . . . . . . .SET THE T/F BIT IN THE RT STATUS WORD
. . . . . . . . . .RESET THE WORD COUNTER
. . . . . . . . . .RESET THE VALUE OF IOLDBSW
. . . . . . . . .ELSE
. . . . . . . . . .NO 0 TO 1 TRANSITIONS IN BIT STATUS WORD
. . . . . . . . . .RESET THE VALUE OF IOLDBSW
. . . . . . . . ENDIF
. . . . . . . . RESET THE HEADER WORD BITS
. . . . . . . . CHECK TO SEE IF THE HEADER WORD HAS CHANGED
. . . . . . . . IF HEADER WORD HAS CHANGED SINCE THE LAST CALL
. . . . . . . . OR TRANSMIT DATA WORDS FLAG IS UP
. . . . . . . . .THEN
. . . . . . . . . .PUT THE DATA WORD BLOCK INTO THE INPUT AR
. . . . . . . . . .DO WHILE I IS LESS THAN TWENTY-NINE
. . . . . . . . . .ENDDO
. . . . . . . . . .PUT THE DATA WORD COUNT ONTO THE RT STATU
. . . . . . . . . .INCREMENT THE WORD COUNTER
. . . . . . . . . .SET TRANSMIT DATA WORDS FLAG TO ZERO
. . . . . . . . . .SET THE DATA SENT FLAG
. . . . . . . . .ELSE
. . . . . . . . . .NO CHANGE IN THE DATA WORDS SO CONTINUE
. . . . . . . . ENDIF
. . . . . . . . CHECK TO SEE IF INPUT ARRAY IS NON-EMPTY OR
. . . . . . . . RT STATUS WORD HAS CHANGED
. . . . . . . . IF NPPWDS IS GREATER THAN ONE OR RT STATUS
. . . . . . . . HAS CHANGED OR TRANSMIT RT STATUS WORD FLAG IS
. . . . . . . . .THEN
. . . . . . . . . .PACK THE NEW WORDS
. . . . . . . . . .CALCULATE NPPWDCT- THE PP WORD COUNT
. . . . . . . . . .CALCULATE NBYTE- THE BYTE COUNT
. . . . . . . . . .ZERO OUT IOADTPP(86)
. . . . . . . . . .PUT THE PP WORD COUNT ONTO THE FIRST BYTE
. . . . . . . . . .PUT THE BYTE COUNT ONTO THE SECOND BYTE
. . . . . . . . . .PACK THE RT STATUS WORD
. . . . . . . . . .CALL THE PACKING ROUTINE
. . . . . . . . . .ZERO OUT THE DATA WORD COUNT
. . . . . . . . . .ZERO OUT THE T/F BIT IN THE RT STATUS WOR
. . . . . . . . . .RESET THE VALUE OF IOLDRT
. . . . . . . . . .SET TRANSMIT RT STATUS WORD FLAG TO ZERO
. . . . . . . . . .RESET THE DATA AVAILABLE FLAG
. . . . . . . . . .SET THE STATUS SENT FLAG
. . . . . . . . .ELSE
. . . . . . . . . .NO CHANGES IN RT WORDS SINCE LAST CSCG CA
. . . . . . . . ENDIF
. . . . . . . ELSE
. . . . . . . . DATA OR STATUS SENT FLAG IS STILL UP
. . . . . . . . "OR" THE VALUES OF STATUS AND DATA SENT
```

```
. . . . . . . . FLAGS ONTO THE APPROPRIATE BIT IN
. . . . . . . . THE ERROR WORDS
. . . . . ENDIF
. . . . ELSE
. . . . . FLAGS ARE NOT EQUAL SO DON'T PACK DATA
. . . . . SET THE SIXTH BIT IN ERROR WORD 1 TO 1
. . . .ENDIF
. . .ELSE
. . . .BIT IS IN PROGRESS SO RETURN
. . ENDIF
. ELSE
. . CSCG IS IN A QUIESCENT STATE SO RETURN
ENDIF
```

```
      SUBROUTINE CSCGNDP
      ABSTRACT
           NORMAL DATA PROCESSING ROUTINE FOR CSCG SUBROUTINE.

      CODING HISTORY
           1.  PROGRAMMED   J. MANGES  CSC  MARCH, 1978
      END OF ABSTRACT
```

```
CHECK FOR DIRECTION OF DATA WORD FLOW
IF K IS EQUAL TO ZERO
. THEN
.  . DATA WORD FLOW IS AYK TO CSCG
.  . READ IN DATA WORD BLOCK FROM THE INPUT BUFFER
.  . DO WHILE L IS LESS THAN 17
.  .  .  .IF IB IS GREATER THAN NUMWDS
.  .  .  .  THEN
.  .  .  .  .  WRAP AROUND TO BEGINNING OF BUFFER
.  .  .  .  ELSE
.  .  .  .  .  CONTINUE TO INCREMENT IB IN A NORMAL MANNER
.  .  .  .ENDIF
.  .  .  .SET BUFFER ELEMENT INTO DATA WORD ARRAY
.  .  .  .ZERO OUT THE BUFFER ELEMENT JUST READ
.  .  .  .FILL UP THE IDATIN ARRAY
.  .  .  .SAVE THE VALUE OF IB
.  . ENDDO
.  . RESET VALUE OF IB
ELSE
.  . DATA FLOW IS CSCG TO AYK
.  . RESET DATA SENT FLAG
ENDIF
```

```
      SUBROUTINE SETABIT(JWORD,NBIT,NUM)
      ABSTRACT
           SETABIT SETS A SPECIFIED BIT TO 0 OR 1 IN A GIVEN WORD
           CALLING PARAMETERS- 1. JWORD- WORD IN WHICH BIT IS TO BE SET
                               2. NBIT- BIT NUMBER OF BIT TO BE RESET
                                  (THE FIRST BIT IN JWORD IS BIT 0)
                               3. NUM-  THE RESET VALUE OF THE BIT

      CODING HISTORY
           1.  PROGRAMMED J. MANGES  12/19/77
      END OF ABSTRACT      .
```

```
      SUBROUTINE READBIT(JWORD,NBIT,NEWWRD)
      ABSTRACT
           READBIT EXTRACTS AND RIGHT JUSTIFIES A GIVEN BIT WITHIN A
           GIVEN WORD.
           CALLING PARAMETERS-
                                1. JWORD- WORD CONTAINING BIT TO BE READ
                                2. NBIT- NUMBER OF BIT TO BE READ
                                   (THE FIRST BIT IN JWORD IS BIT 0)
                                3.  NEWWRD- RIGHT JUSTIFIED RETURN VALUE
                                          OF BIT.
      CODING HISTORY
           1.  PROGRAMMED  J. MANGES  CSC  12/19/77
      END OF ABSTRACT
```

```
      SUBROUTINE PACKPP(NRT,NPPWDS)
      ABSTRACT
           PACKPP WRITES OUT THE HEADER WORD IN THE OUTPUT ARRAY
           AND THE NON-ZERO CONTENTS OF THE INPUT ARRAY.

      CODING HISTORY
           1.  PROGRAMMED  J. MANGES CSC  DEC. 1977
      END OF ABSTRACT
```

```
      DO WHILE I IS LESS THAN NPPWDS+200
      ENDDO
```

```
      SUBROUTINE SONOINF
      ABSTRACT
           THIS ROUTINE OUTPUTS SONOBUOY RECEIVER UNIT CHANNEL SELECTION
           TO THE SONOBOY ROUTINE.
      CODING HISTORY
           1.  PROGRAMMED  J. MANGES  CSC  MARCH 1978

      END OF ABSTRACT
```

```
      OUTPUT SONOBUOY RECEIVER UNIT CHANNEL SELECTIONS TO SONOBOY ROUTIN
      OBTAIN SWITCH FUNCTION VALUES FROM DATA WORDS
      TRANSLATE THE SWITCH FUNCTION VALUES INTO CHANNEL NUMBERS
      AND INSERT INTO THE IRFCH ARRAY
      DO WHILE K IS LESS THAN EIGHT
      .  . DO WHILE I IS LESS THAN 32
      .  .  .  .IF NRCVR(K) EQUALS ICHANL(I)
      .  .  .  .  THEN
      .  .  .  .  .  PLACE THE CHANNEL NUMBER INTO THE IRFCH ARRAY
      .  .  .  .  .  CONTINUE THE PROCESS FOR THE NEXT RECEIVER UNIT
      .  .  .  .  .  GO TO 200
      .  .  .  .  ELSE
      .  .  .  .  .  CONTINUE TO LOOK FOR THE CHANNEL NUMBER
      .  .  .  .  .  CORRESPONDING TO THE SWITCH FUNCTION VALUE
      .  .  .  .ENDIF
      .  . ENDDO
      ENDDO
```

```
      SUBROUTINE HEADER
      ABSTRACT
           THIS SUBROUTINE SETS BITS IN THE OUTGONG HEADER DATA WORD TO
           INDICATE CHANGES IN DATA WORDS SINCE THE LAST CSCG CALL.

      CODING HISTORY
           1.  PROGRAMMED  J. MANGES  CSC  MARCH 1978
      CHECK TO SEE WHICH OUTGOING DATA WORDS HAVE CHANGED SINCE THE LAST
      CSCG CALL
      DO WHILE I IS BETWEEN TWO AND THIRTEEN
           IF IDATOUT(I) HAS CHANGED
             THEN
                SET THE APPROPRIATE HEADER WORD BIT TO INDICATE A CHANGE
             ELSE
                SET THE APPROPRIATE HEADER WORD BIT TO INDICATE NO CHANG
           ENDIF
      ENDDO
      DO WHILE I IS LESS THAN 2
           IF IDATOUT(I+13) OR IDATOUT(I+14) HAS CHANGED
             THEN
               SET THE HEADER WORD BIT TO INDICATE A CHANGE
             ELSE
               SET THE HEADER WORD BIT TO INDICATE NO CHANGE
           ENDIF
      ENDDO
      CHECK FOR CHANGES IN THE REMAINING DATA WORDS NOT COVERED BY
      THE HEADER WORD
      DO WHILE L IS LESS THAN TWENTY-NINE
           IF IDATOUT(L) HAS CHANGED
             THEN
                SET FLAG TO TRANSMIT THE DATA WORDS
             ELSE
                CONTINUE TO LOOK FOR DATA WORD CHANGES
           ENDIF
      ENDDO
      RESET THE VALUES OF THE OLD DATA WORDS
      RESET THE VALUE OF IOLDAT(I)
      DO WHILE J IS LESS THAN 29
      ENDDO
```

```
      SUBROUTINE PERIPHL

      ABSTRACT
           PERIPHL UPDATES THE SONOBUOY RECEIVER SIGNAL STRENGTH AND THE
           CURRENT D/L MODE SETTING AND SETS THE APPROPRIATE DATA WORD B

      CODING HISTORY
           1.  PROGRAMMED  J. MANGES   CSC  MARCH 1978
      END OF ABSTRACT
```

```
SET THE CONSTANT C=DMAX/7 WHERE DMAX IS THE DISTANCE AT WHICH
SONOBUOY SIGNAL STRENGTH JUST REACHES ZERO
UPDATE PERIPH EQUIPMENT OPERATING STATUS
CHECK STATUS OF DATA LINK MODE  (ASW OR ASHD)
IF D/L MODE IS ASW
. THEN
.  . SET BIT 0 IN IDATOUT(18) TO 1
. ELSE
.  . SET BIT 0 IN IDATOUT(18) TO 0
ENDIF
UPDATE SONO RCVRS 1,2 SIGNAL STRENGTH
LOOP THRU FOR EIGHT SONOBUOY RECEIVER UNITS
DO WHILE K IS LESS THAN EIGHT
.  . CALCULATE THE DISTANCE FROM HELO TO BUOY ASSIGNED TO RCVR UNI
.  . ASSIGN A NUMBER (0-7) TO THE SIGNAL STRENGTH FOR EACH RECEIVE
.  . DO WHILE N IS LESS THAN SEVEN
.  .  .  .IF DIST(K) GE (N-1)*C AND LT N*C
.  .  .  .  THEN
.  .  .  .  .  ASSIGN SIGNAL STRENGTH LEVEL 8-N TO DIST(K)
.  .  .  .  ELSE
.  .  .  .  .  CONTINUE LOOPING THROUGH DISTANCE INTERVALS
.  .  .  .ENDIF
.  . ENDDO
.  . SINCE LOOP IS EXHAUSTED DIST(K) IS SO LARGE THAT ISIGSTR(K)=0
ENDDO
ZERO OUT STATUS WORD 18
INSERT SIGNAL STRENGTH FOR RECEIVER UNITS A,B,E,F
ZERO OUT STATUS WORD 19
INSERT SIGNAL STRENGTH FOR RECEIVER UNITS C,D,G,H
```

```
      SUBRCUTINE UDICP
      ABSTRACT
           THIS SUBROUTINE DETECTS CHANGES IN SWITCH SETTINGS FOR UHF
           MODE AND CHANNEL SELECTION AND RESETS THE APPROPRIATE BITS
           IN THE OUT-GOING DATA WORDS.

      CODING HISTORY
           1.  PROGRAMMED-  J. MANGES   CSC   MARCH 1978

      END OF ABSTRACT      .
```

```
      UPDATE UHF-1 MODE SELECTION (OTPI OR ADF)
      CHECK MODE SELECT SWITCH SETTING
      RESET DATA WORD BITS IN STATUS WORD NO. 20
      UPDATE UHF-1 CHANNEL SELECTION
      BLANK OUT THE CHANNEL SELECTION FIELD IN IDATOUT(13)
      CHECK UHF-1 CHANNEL SELECT UNITS SWITCH SETTING AND COPY IT
      INTO THE DATA WORD
      CHECK THE UHF-1 CHANNEL SELECT TENS SWITCH SETTING AND COPY IT
      INTO THE DATA WORD
```

```
      SUBROUTINE DTOAINF
      ABSTRACT

           THIS ROUTINE OUTPUTS OTPI BEARING TO THE DTOA SUBROUTINE.
      CODING HISTORY
           1.  STOLEN FROM HELONAV SUBROUTINE  MAY, 1978

      END OF ABSTRACT
```

```
      CHECK UHF-1 MODE
      IF UHF IS IN OTPI MODE
      . THEN
      .  . READ RF CHANNEL SELECTION FROM DISCRETES
      .  . FIND A LAUNCHED AND ACTIVE BUOY TUNED TO THE ABOVE RF CHANNEL
      .  . LOOP THRU FOR 32 BUOYS
      .  . DO WHILE I IS LESS THAN 32
      .  .  .  .IF BUOY IS STILL ACTIVE AND IN THE WATER
      .  .  .  .THEN
      .  .  .  .  .  IF TIME TO PING AND TIME SINCE LAST PING OK
      .  .  .  .  .  THEN
      .  .  .  .  .  .  . IF BUOY IS CASS MAKE SURE VHF TRANS. IS ON
      .  .  .  .  .  .  . THEN
      .  .  .  .  .  .  .  .  .IF SWITCH SETTING COINCIDES WITH BUOYRF N
      .  .  .  .  .  .  .  .  .THEN
      .  .  .  .  .  .  .  .  .  .  SET THE CHUTE NUMBER OF THE BUOY PRE
      .  .  .  .  .  .  .  .  .  .  TUNED TO THE UHF RADIO
      .  .  .  .  .  .  .  .  .  .  CALCULATE THE OTPI BEARING
      .  .  .  .  .  .  .  .  .ELSE
      .  .  .  .  .  .  .  .  .  .  CONTINUE TO LOOP THRU THE BUOYS
      .  .  .  .  .  .  .  .  .ENDIF
      .  .  .  .  .  .  .  . ELSE
      .  .  .  .  .  .  .  .  .BUOY IS CASS-TYPE AND VHF TRANS. NOT ON
      .  .  .  .  .  .  .  . ENDIF
      .  .  .  .  .  . ELSE
      .  .  .  .  .  .  . OVER 20 SEC SINCE LAST PING AND NO NEW PING YE
      .  .  .  .  . ENDIF
      .  .  .  .ELSE
      .  .  .  .  . BUOY NOT STILL ACTIVE OR NOT IN THE WATER
      .  .  .  .ENDIF
      .  . ENDDO
      .  . LOOP IS EXHAUSTED-NO BUOY TUNED TO UHF
      . ELSE
      .  . UHF NOT IN OTPI MODE SO NO OTPI BEARING DETERMINATION
      ENDIF
```

MULTIFUNCTION CONTROL SET MODULE

(MFCS)

```
      DO UNTIL THERE IS NO MORE DATA
      .  . DO UNTIL ALL ID POSSIBILITIES HAVE BEEN TESTED
      .  .  .  .IF ID MATCHES
      .  .  .  .  THEN
      .  .  .  .  .  SET DATA TYPE INDEX AND EXIT LOOP
      .  .  .  .  ELSE
      .  .  .  .ENDIF
      .  . ENDDO
      .  . CASE DATA TYPE (IDX)
      .  .  .  .*IDX .EQ. 1
      .  .  .  .INDISCRETES
      .  .  .  .*IDX .EQ. 2
      .  .  .  .DATA FROM AYK
      .  .  .  .*IDX .EQ. 3
      .  .  .  .NEW BIT STATUS
      .  .  .  .*IDX .EQ. 4
      .  .  .  .RT STATUS WORD CHANGE
      .  .  .  .*IDX .EQ. 5
      .  .  .  .PP DATA AVAIL  BIT CHANGED
      .  . END CASE
      END DO
      RESET OUTDISCRETE ARRAY *IOUT*
      CALL THE MFCS MODULE TO PROCESS THE CURRENT DATA SET
      PRINT MFCS' OUTPUT BUFFERS AND FLAGS
      DO UNTIL ALL *NIND* WORDS ARE RESET TO ZERO
      END DO
```

```
DO UNTIL ALL WORDS HAVE BEEN EXPANDED
.  . DO UNTIL ALL BITS HAVE BEEN EXPANDED FOR A GIVEN WORD
.  . END DO
END DO
```

```
      SUBROUTINE MFCS
      ABSTRACT
           THIS ROUTINE PROCESSES SWITCH CLOSURE DATA FOR THE
           MULTIFUNCTION CONTROL SETS, ATO AND SO. DATA IS TRANSFERRED
           BETWEEN THE SETS AND THE AOP.

           CODING HISTORY
                1. PROGRAMMED -- ROBERT J. HUBER  NOVEMBER 1977 (CSC)
      END OF ABSTRACT
```

```
DO WHILE IKEYSET EQUAL FROM ONE(AFO) TO TWO (SO)
. . RESET OUTPUT BUFFER WORD COUNTER AND BUFFER WORD ONE
. . SET INPUT BUFFER POINTER
. . LOOP THRU AYK INPUT BUFFER UNTIL A WORD WITH ZERO
. . IS INCOUNTERED -- BUFFER EMPTY INDICATOR
. . IF THE I-TH WORD OF BUFFER NOT ZERO
. .  .THEN
. . .  .DECODE AOP COMMAND WORD
. . .  .IF MODE/DISCRETE DATA COMMAND
. . .  .  THEN
. . .  .  .  DETERMINE TYPE OF DATA
. . .  .  .  IF DATA WORD COUNT IS ONE
. . .  .  .  . THEN
. . .  .  .  . . INITIALIZE KEYSET
. . .  .  .  . ELSE
. . .  .  .  . . TEST FOR INITIATE PROCESSING
. . .  .  .  . . IF DATA WORD COUNT IS FOUR
. . .  .  .  . . .THEN
. . .  .  .  . . . .INITIATE PROCESSING
. . .  .  .  . . .ELSE
. . .  .  .  . . . .TEST FOR INITIATE SELF-TEST
. . .  .  .  . . . .IF DATA WORD COUNT IS THREE
. . .  .  .  . . . .  THEN
. . .  .  .  . . . .  .  INITIATE SELF-TEST MODE
. . .  .  .  . . . .  .  SET RECEIVE BUSY BIT
. . .  .  .  . . . .  ELSE
. . .  .  .  . . . .  .  COMMAND NOT PROCESSED BY KEYSETS
. . .  .  .  . . . .  .ENDIF
. . .  .  .  . . .ENDIF
. . .  .  .  ENDIF
. . .  .  ELSE
. . .  .  .  TEST FOR NORMAL DATA TRANSFER
. . .  .  .  . THEN
. . .  .  .  . . PROCESS NORMAL DATA TRANSFER
. . .  .  .  . . IF RECEIVE/TRANSMIT FLAG IS ZERO
. . .  .  .  . . .THEN
. . .  .  .  . . . .RECEIVE DATA (CALL MFCSNDX)
. . .  .  .  . . .ELSE
. . .  .  .  . . . .TRANSMIT DATA TO AYK
. . .  .  .  . . . .RESET OUTPUT BUFFER FULL FLAG
. . .  .  .  . . .ENDIF
. . .  .  .  . .ELSE
. . .  .  .  . . COMMAND NOT PROCESSED BY KEYSETS
. . .  .  .  ENDIF
. . .  .ENDIF
. . .  .INCREMENT BUFFER WORD COUNT POINTER
. . .  .AND ZERO WORD JUST PROCESSED IN IRTBUFF ARRAY
. .  .ELSE
. . ENDIF
. . CHECK QUIESCENT AND SELF-TEST FLAGS
. . IF KEYSET IN QUIESCENT STATE
. .  .THEN
. . .  .SET RETURN FLAG
. .  .ELSE
. . .  .TEST FOR SELF-TEST IN PROGRESS
. . .  .  THEN
```

```
. . . . . DECREMENT COUNTER BY 200 MSEC
. . . . . IF SELF-TEST NOT COMPLETED
. . . . . . THEN
. . . . . . . SET RETURN FLAG
. . . . . . ELSE
. . . . . ENDIF
. . . . ELSE
. . . .ENDIF
. . .ELSE
. . ENDIF
. . IF RETURN FLAG IS NOT SET
. . .THEN
. . . .PROCESS NORMAL KEYSET
. . .ELSE
. . . .RETURN
. . ENDIF
. . SET KEYSET INPUT BUFFER COUNTER
ENDDO
```

```
      SUBROUTINE MFCSNDX
      ABSTRACT
           THIS ROUTINE PROCESSES SWITCH DATA FROM THE AYK TO
           THE SIMULATION HARDWARE KEYSETS

      CODING HISTORY
           1. PROGRAMMED ROBJERT J. HUBER     NOVEMBER 1977  (CSC)

      END OF ABSTRACT
```

```
DETERMINE WHICH KEYSET THE DATA IS FOR
IF IT'S THE ATO KEYSET
. THEN
.   . BREAK DOWN SWITCH LIGHTING DATA FOR ATO KEYSET
.   . ZERO FIRST FIVE WORDS OF DATA TRANSFER AND
.   . THE COMMAND WORD THEN INCREMENT BUFFER POINTER
.   . DO WHILE THERE ARE WORDS TO BE ZEROED
.   .   .  .IF POINTER AT WRAP AROUND POINT
.   .   .  .  THEN
.   .   .  .  .  RESET POINTER TO *1*
.   .   .  .  ELSE
.   .   .  .  .  POINTER NOT RESET
.   .   .  .ENDIF
.   . ENDDO
.   . DO WHILE I IS A VALID ATO SWITCH NUMBER (0 THRU 73)
.   .   .  .IF INPUT BUFFER POINTER CHANGED
.   .   .  .  THEN
.   .   .  .  .  ZERO PREVIOUS BUFFER WORD
.   .   .  .  ELSE
.   .   .  .  .  TAKE NO ACTION
.   .   .  .ENDIF
.   .   .  .IF POINTER AT WRAPAROUND POINT
.   .   .  .  THEN
.   .   .  .  .  RESET POINTER TO *1*
.   .   .  .  ELSE
.   .   .  .  .  POINTER NOT RESET
.   .   .  .ENDIF
.   . ENDDO
.   . ZERO FINAL BUFFER WORD
. ELSE
.   . INSERT SWITCH LIGHTING DATA FOR THE SO KEYSET
.   . SWITCH INFO FOR THE SO KEYSET IS SEQUENCIAL IN
.   . THE *IOUT* ARRAY
.   . DO UNTIL NEXT TEN INPUT WORDS LOCATED AND SAVED
.   .   .  .IF POINTER AT WRAP AROUND POINT
.   .   .  .  THEN
.   .   .  .  .  RESET POINTER TO *1*
.   .   .  .  ELSE
.   .   .  .  .  POINTER NOT RESET
.   .   .  .ENDIF
.   . ENDDO
.   . DO UNTIL INPUT BUFFER IS ZEROED FOR THIS TRANSFER
.   . END DO
ENDIF
```

```
      SUBROUTINE MFCSPRC
      ABSTRACT

           THIS ROUTINE PROCESSES SWITCH CLOSURE DATA FROM THE ATO AND
           SO KEYSETS TO BE SENT TO THE AYK-14 AOP

      PROGRAMMING HISTORY

           1. PROGRAMMED -- ROBERT J. HUBER NOVEMBER 1977 (CSC)

      END OF ABSTRACT
```

```
IF ATO FAULT BIT SET HIGH
. THEN
.  . SET UP BIT TO BE TRANSMITTED TO AOP
. ELSE
.  . SET OLD BIT TO NEW BIT
ENDIF
CHECK FOR SW CLOSURE DATA TO BE SENT TO AOP
IF THERE IS KEYSWITCH DATA
. THEN
.  . SETUP OUTPUT BUFFER TO AOP
. ELSE
ENDIF
IF THERE IS DATA TO BE OUTPUT
. THEN
.  . IF THE CP AND PP DATA AVAIL FLAGS EQUAL
.  .  .THEN
.  .  .  .IF STATUS SENT FLAG CLEAR
.  .  .  .  THEN
.  .  .  .  .  IF DATA SENT FLAG CLEAR
.  .  .  .  .  . THEN
.  .  .  .  .  .  . SETUP OUTPUT BUFFER FOR AOP
.  .  .  .  .  .  . RESET OUTPUT BUFFER WORD COUNTER
.  .  .  .  .  .  . SET STATUS SENT FLAG
.  .  .  .  .  .  . IF DATA HAS BEEN SENT
.  .  .  .  .  .  .  .THEN
.  .  .  .  .  .  .  .  .SET DATA SENT FLAG
.  .  .  .  .  .  .  .  .ELSE
.  .  .  .  .  .  .  . ENDIF
.  .  .  .  .  .  .  . COMPLEMENT CP DATA AVAIL FLAG
.  .  .  .  .  .  .  . SET OLD BIT TO NEW BIT
.  .  .  .  .  .  . ELSE
.  .  .  .  .  .  .  . SET DATA SENT ERROR FLAG
.  .  .  .  .  .  ENDIF
.  .  .  .  .  ELSE
.  .  .  .  .  .  SET STATUS SENT ERROR FLAG
.  .  .  .ENDIF
.  .  .ELSE
.  .  .  .SET PP ERROR FLAG
ENDIF
.  . ENDIF
. ELSE
.  . NO OUTPUT TO AOP THIS CYCLE
```

```
      SUBROUTINE MFCSIND
      ABSTRACT

           THIS ROUTINE PROCESSES DISCRETES FROM THE KEYSETS
           (ATO AND SO) TO BE SENT TO THE AYK-14 AOP

      CODING HISTORY

           1. PROGRAMMED -- ROBERT J. HUBER NOVEMBER 1977 (CSC)

      END OF ABSTRACT
```

```
IF THE SO KEYSET IS BEING PROCESSED
. THEN
.  . DECODE SO DISCRETES
.  . TRANSFER SO INDISCRETES INTO WORKING ARRAY
.  . DO UNTIL ALL INDISCRETES ARE SCANNED
.  .  .  .IF SO INDISCRETE HAS *HI* BIT(S)
.  .  .  .  THEN
.  .  .  .  .  DO UNTIL ALL BITS ARE TESTED
.  .  .  .  .  .  . IF THE J-TH BIT IS *HI*
.  .  .  .  .  .  .  .THEN
.  .  .  .  .  .  .  .  .INCREMENT WORD COUNT FOR RT STATUS
.  .  .  .  .  .  .  .  .AND PUT KEY VALUE INTO HOLDING
.  .  .  .  .  .  .  .  .ARRAY
.  .  .  .  .  .  .  .  .IF DATA WORD COUNT IS TWO
.  .  .  .  .  .  .  .  .  THEN
.  .  .  .  .  .  .  .  .  .  SET INDIS TO ZERO
.  .  .  .  .  .  .  .  .  ELSE
.  .  .  .  .  .  .  .  .ENDIF
.  .  .  .  .  .  .  .ELSE
.  .  .  .  .  .  . ENDIF
.  .  .  .  .  ENDDO
.  .  .  .  ELSE
.  .  .  .ENDIF
.  . ENDDO
. ELSE
.  . PROCESS ATO KEYSWITCH CLOSURES
.  . TRANSFER INDISCRETES INTO WORKING ARRAY FOR ATO
.  . DO WHILE THERE ARE KEYWORDS FOR THE ATO KEYSET
.  .  .  .IF ANY OF THE BITS ARE SET IN THE INDISCRETE WORD
.  .  .  .  THEN
.  .  .  .  .  EXTRACT SWITCH CLOSURE DATA
.  .  .  .  .  DO WHILE J IS ONE OF THE SIXTEEN BITS OF DATA
.  .  .  .  .  . IF THE BIT IS ON
.  .  .  .  .  .  .THEN
.  .  .  .  .  .  .  .IF IT'S AN ATO KEY SWITCH
.  .  .  .  .  .  .  .  THEN
.  .  .  .  .  .  .  .  .  INCREMENT WORD COUNT FOR RT STATUS
.  .  .  .  .  .  .  .  .  WORD AND HOLDING ARRAY,THEN PUT
.  .  .  .  .  .  .  .  .  SWITCH NUMBER INTO HOLDING ARRAY
.  .  .  .  .  .  .  .  ELSE
.  .  .  .  .  .  .  .ENDIF
.  .  .  .  .  .  .  .ELSE
.  .  .  .  .  .  . ENDIF
.  .  .  .  .  ENDO
.  .  .  .  ELSE
.  .  .  .ENDIF
.  . ENDDO
ENDIF
```

MAD SIGNAL PROCESSOR MODULE

(MSP)

```
      PROGRAM MSPDRIV
      ABSTRACT
           THIS PROGRAM IS A DRIVER TEST PROGRAM FOR THE MSP MODULE.

      CODING HISTORY
           1. PROGRAMMED--ALEX PODLECKI        11/03/77

      END OF ABSTRACT
```

```
ZERO INPUT BUFFER
DOWHILE ANOTHER INPUT BUFFER WORD AVAILABLE
. . INSERT ZERO WORD
ENDDO
LOCKON DETECT (SYSTEM READY) SET
ALTITUDE = 512 FEET
PRELIMINARY EVENT FLAG ON
TIME OF PRELIMINARY EVENT = 100 SECONDS
CURRENT TIME = 101 SECONDS
R/V = 1.0
CONFIRMED DETECT FLAG ON FOR 12,*)
TIME OF CONFIRMED DETECT = 100 SECONDS
SLANT RANGE = 7 FEET
DEGREES TO RADIANS
MSB GAMMAS = 18,  LSB GAMMAS = 18
ROLL = 22.5 DEGREES
HEADING = 11.25 DEGREES
GROUND SPEED = 64 KNOTS
LONGITUDE = 180 BAMS
LATITUDE = 5.625 BAMS
. . . = 87 DEGREES 10.95 MINUTES (SOUTH)
CHANGE IN POSITION
DELTA X = +1 FOOT   DELTA Y = -1 FOOT
NO INITIAL FAULTS
INITIALIZE CP,PP DATA AVAILABLE WORDS
RESPONSE TO NO COMMANDS
IPL SEQUENCE
1. CONTROL COMMAND
2. MULTI-MESSAGE TRANSFER
3. NORMAL DATA TRANSFER
DOWHILE ADDITIONAL IPL WORDS REQUIRED
. . INSERT DUMMY IPL WORD IN INPUT BUFFER
ENDDO
4. NULL
SELF-TEST SEQUENCE
DO WHILE RT SAYS ITS BUSY
. . EXECUTE MSP
ENDDO
INITIALIZATION SEQUENCE
1. INITIALIZE TERMINAL
2. TRANSMIT BIT STATUS
3. INITIATE PROCESSING
ALSO EXECUTE WITH NO INPUT
DATA TRANSFER SEQUENCE
9 WORDS TO BE TRANSFERRED
WORD 1 - ALTITUDE = 512 FEET
WORD 2 - SPEED = 64 KNOTS
WORD 3 - HEADING = 11.25 DEGREES
WORD 4 - ROLL = 22.5 DEGREES
WORD 5 - MAD MSB = 18
WORD 6 - MAD LSB = 18
WORD 7 - POSITION = 5.625 BAMS LATITUDE, 180 BAMS LONGITUDE
WORD 8 - ALTITUDE COMPENSATION = -7.0 FEET
WORD 9 - OPTION = 0
SEND DATA TO MSP
SET UP REQUEST FOR DATA FROM MSP
```

```
SEND REQUEST TO MSP
SET UP CHANGE OF OPTION TO 1
1 WORD TO BE TRANSFERRED
WORD 1 - OPTION = 1
SEND DATA TO MSP
SET UP CHANGE OF OPTION TO 2
1 WORD TO BE TRANSFERRED
WORD 1 - OPTION = 2
SEND DATA TO MSP
SEND REQUEST TO SEND DATA
SET UP NULL INPUT
SEND TO MSP
ERROR PROCESSING SECTION
SEND NULL INPUT WITH ERROR BIT ACTIVATED
END OF PROGRAM
```

```
      SUBROUTINE XMSP(IN,LWA)
      ABSTRACT
           THIS ROUTINE PERFORMS THREE FUNCTIONS

           1. IT PRINTS CONTENTS OF CURRENT INPUT BUFFER
           2. IT EXECUTES THE MSP MODULE
           3. IT PRINTS THE CONTENTS OF THE RESULTANT OUTPUT BUFFER
              AND RESETS THE PP BIT TO SIGNIFY ACCEPTANCE.

           IN - POINTER TO FIRST WORD OF INPUT

           LWA - POINTER TO LAST WORD ADDRESS OF INPUT BUFFER
           (NOTE - INPUT BUFFER IS CIRCULAR TERMINATED BY ZERO WORD)

      CODING HISTORY
           1. PROGRAMMED--ALEX PODLECKI        11/07/77

      END OF ABSTRACT
```

```
PRINT INPUT BUFFER
DOWHILE SOMETHING IN INPUT BUFFER
.  . EXPAND INPUT WORD
.  . PRINT INPUT WORD BIT-BY-BIT
ENDDO
IF EMPTY INPUT BUFFER AND PRINT MESSAGES SELECTED
. THEN
.  . PRINT INFORMATIVE MESSAGE
. ELSE
.  . OMIT MESSAGE
ENDIF
EXECUTE THE MSP MODULE
PRINT OUTPUT BUFFER
IF OUTPUT BUFFER EMPTY
. THEN
.  . IF PRINT MESSAGES SELECTED
.  .  .THEN
.  .  .  .PRINT INFORMATIVE MESSAGE
.  .  .ELSE
.  .  .  .OMIT MESSAGE
.  . ENDIF
. ELSE
.  . PRINT HEADER WORD
.  . DOWHILE SOMETHING IN OUTPUT
.  .  .  .EXPAND OUTPUT WORD
.  .  .  .PRINT OUTPUT WORD BIT-BY-BIT
.  . ENDDO
.  . SET PP BIT
ENDIF
IF ERROR BIT SET OR PRINT MESSAGES SELECTED
. THEN
.  . PRINT ERROR STATUS
. ELSE
.  . OMIT MESSAGE
ENDIF
IF ERROR WORD BIT IS SET
. THEN
.  . PRINT THE ERROR WORD
. ELSE
.  . DO NOT PRINT THE ERROR WORD
ENDIF
END OF ROUTINE
```

```
      SUBROUTINE ADVANCE(POINTER,LWA)
      ABSTRACT
           THIS ROUTINE INCREMENTS A POINTER BY 1. IF THE POINTER WAS
           ALREADY SET TO AN LWA, THE POINTER IS RESET TO 1.

           POINTER - CURRENT VALUE OF POINTER

           LWA - LAST WORD ADDRESS FOR POINTER

      CODING HISTORY
           1. PROGRAMMED--ALEX PODLECKI        11/04/77

      END OF ABSTRACT
```

```
IF POINTER IS LESS THAN LWA
. THEN
.   . INCREMENT POINTER
. ELSE
.   . SET POINTER TO FWA
ENDIF
```

```
      SUBROUTINE EXPAND(N,IN,OUT)
      ABSTRACT
           THIS ROUTINE EXPANDS A WORD INTO AN N WORD ARRAY SUCH THAT
           WORD 1 CONTAINS BIT N-1, WORD 2 CONTAINS BIT N-2, ...,
           AND WORD N CONTAINS BIT 0 ( RIGHT JUSTIFIED WITH ZERO FILL )

           N - NUMBER OF BITS TO BE EXPANDED

           IN - INPUT WORD TO BE EXPANDED

           OUT - OUTPUT ARRAY TO RECEIVE EXPANSION

      CODING HISTORY
           1. PROGRAMMED--ALEX PODLECKI        11/07/77

      END OF ABSTRACT
```

```
DOWHILE ANOTHER BIT TO BE EXPANDED
.  . MASK OUT DESIRED BIT
.  . SET UP FOR NEXT BIT
ENDDO
```

```
      SUBROUTINE PACKPP( NRT, N)
      ABSTRACT
           THIS ROUTINE IS A DUMMY SUBSTITUTE FOR THE ACTUAL PACKPP.

           NRT - NUMBER CORRESPONDING TO RT

           N - NUMBER OF WORDS TO BE *PACKED*

      CODING HISTORY
           1. PROGRAMMED--ALEX PODLECKI        11/07/77

      END OF ABSTRACT
```

EXIT

```
      SUBROUTINE MSP
      ABSTRACT
           THIS ROUTINE SENDS EVENT AND CONTACT MESSAGES TO AN/AYK-14
           FROM MADMOD DATA AND VERIFIES ALTITUDE, SPEED, HEADING
           AND ROLL.

      CODING HISTORY
           1. PROGRAMMED--ALEX PODLECKI        10/26/77

      END OF ABSTRACT
```

```
MAIN LOOP FOR INPUT PROCESSING
IF AN/AYK-14 COMMAND
. THEN
.  . RESET STATUS SENT FLAG
. ELSE
.  . DO NOT RESET STATUS SENT FLAG
ENDIF
DO WHILE SOMETHING IN INPUT BUFFER
.  . CRACK COMMAND WORD INTO BASIC FIELDS
.  . IF POINTER IS LESS THAN LWA
.  .  .THEN
.  .  .  .INCREMENT POINTER
.  .  .ELSE
.  .  .  .SET POINTER TO FWA
.  . ENDIF
.  . IF COMMAND IS A MODE/DISCRETE
.  .  .THEN
.  .  .  .PROCESS MODE/DISCRETES
.  .  .  .IF MODE/DISCRETE IS INITIALIZE RT
.  .  .  .  THEN
.  .  .  .  .  INITIALIZE RT
.  .  .  .  ELSE
.  .  .  .  .  CONTINUE PROCESSING MODE/DISCRETES
.  .  .  .ENDIF
.  .  .  .IF MODE/DISCRETE IS INITIATE SELF-TEST
.  .  .  .  THEN
.  .  .  .  .  INITIATE SELF-TEST
.  .  .  .  ELSE
.  .  .  .  .  CONTINUE PROCESSING MODE/DISCRETES
.  .  .  .ENDIF
.  .  .  .IF MODE/DISCRETE IS INTIATE PROCESSING
.  .  .  .  THEN
.  .  .  .  .  INITIATE PROCESSING
.  .  .  .  ELSE
.  .  .  .  .  CONTINUE PROCESSING MODE/DISCRETES
.  .  .  .ENDIF
.  .  .  .OTHER MODE/DISCRETES ARE NO-OPS
.  .  .ELSE
.  .  .  .CONTINUE PROCESSING OTHER COMMANDS
.  . ENDIF
.  . IF COMMAND IS A NORMAL DATA TRANSFER
.  .  .THEN
.  .  .  .PROCESS NORMAL DATA TRANSFERS
.  .  .  .IF BC IS REQUESTING DATA
.  .  .  .  THEN
.  .  .  .  .  IIU HAS TRANSFERRED PREVIOUS OUTPUT
.  .  .  .  .  RESET DATA SENT FLAG
.  .  .  .  ELSE
.  .  .  .  .  DO NOT RESET DATA SENT FLAG
.  .  .  .  .  BC IS SENDING DATA
.  .  .  .  .  IF IPL IS IN PROGRESS
.  .  .  .  .  . THEN
.  .  .  .  .  .  . IPL IS BEING TERMINATED
.  .  .  .  .  .  . DO WHILE IPL WORDS IN BUFFER
.  .  .  .  .  .  .  .  .IGNORE IPL WORDS
.  .  .  .  .  .  .  .  .IF *IN* IS LESS THAN *LWA*
```

```
. . . . . . . . . THEN
. . . . . . . . .  INCREMENT *IN*
. . . . . . . . . ELSE
. . . . . . . . .  SET *IN* TO *FWA*
. . . . . . . . . .ENDIF
. . . . . . . . ENDDO
. . . . . . . . IF OBSERVED WORD COUNT MATCHES CONTROL COMMAND
. . . . . . . .  .THEN
. . . . . . . .  .NO MULTI-MESSAGE WORD COUNT ERROR
. . . . . . . .  .ELSE
. . . . . . . .  .MULTI-MESSAGE WORD COUNT ERROR
. . . . . . . . ENDIF
. . . . . . . ELSE
. . . . . . . . INPUT DATA IS TO BE PROCESSED
. . . . . . . . DO WHILE INPUT DATA IN BUFFER
. . . . . . . . .  .CRACK IDENT FIELD FOR DATA WORD
. . . . . . . . .  .CASE MODE VALUE (IDENT)
. . . . . . . . .  .GO TO (400,410,420,430,440,450,460,470)
. . . . . . . . .  .  IDENT + 1
. . . . . . . . .  *IDENT EQ 0
. . . . . . . . .  DECODE ALTITUDE
. . . . . . . . .  IF WITHIN TOLERANCE
. . . . . . . . .  . THEN
. . . . . . . . .  . . NO ERROR
. . . . . . . . .  . ELSE
. . . . . . . . .  . . ERROR
. . . . . . . . .  ENDIF
. . . . . . . . .  SET TOLERANCE ERROR STATUS
. . . . . . . . .  *IDENT EQ 1
. . . . . . . . .  DECODE GROUND SPEED
. . . . . . . . .  IF WITHIN TOLERANCE
. . . . . . . . .  . THEN
. . . . . . . . .  . . NO ERROR
. . . . . . . . .  . ELSE
. . . . . . . . .  . . ERROR
. . . . . . . . .  ENDIF
. . . . . . . . .  SET TOLERANCE ERROR STATUS
. . . . . . . . .  *IDENT EQ 2
. . . . . . . . .  DECODE HEADING
. . . . . . . . .  IF WITHIN TOLERANCE
. . . . . . . . .  . THEN
. . . . . . . . .  . . NO ERROR
. . . . . . . . .  . ELSE
. . . . . . . . .  . . ERROR
. . . . . . . . .  ENDIF
. . . . . . . . .  SET TOLERANCE ERROR STATUS
. . . . . . . . .  *IDENT EQ 3
. . . . . . . . .  DECODE ROLL
. . . . . . . . .  IF ROLL IS NEGATIVE
. . . . . . . . .  . THEN
. . . . . . . . .  . . CONVERT TO 1'S COMPLEMENT
. . . . . . . . .  . . EXTEND SIGN AND EVALUATE
. . . . . . . . .  . ELSE
. . . . . . . . .  . . EVALUATE AS IS
. . . . . . . . .  ENDIF
. . . . . . . . .  IF WITHIN TOLERANCE
```

```
. . . . . . . . . . . . . THEN
. . . . . . . . . . . . .   . NO ERROR
. . . . . . . . . . . . . ELSE
. . . . . . . . . . . . .   . ERROR
. . . . . . . . . . . . . ENDIF
. . . . . . . . . . . . . SET TOLERANCE ERROR STATUS
. . . . . . . . . . . . . *IDENT EQ 4
. . . . . . . . . . . . . DECODE 1ST WORD OF TOTAL MAD FIELD
. . . . . . . . . . . . . IF *IN* IS LESS THAN *LWA*
. . . . . . . . . . . . . . THEN
. . . . . . . . . . . . .   . INCREMENT *IN*
. . . . . . . . . . . . . . ELSE
. . . . . . . . . . . . .   . RESET *IN* TO *FWA*
. . . . . . . . . . . . . ENDIF
. . . . . . . . . . . . . DECODE 2ND WORD OF TOTAL MAD FIELD
. . . . . . . . . . . . . *IDENT EQ 5
. . . . . . . . . . . . . OBTAIN CURRENT POSITION
. . . . . . . . . . . . . CONVERT LATITUDE TO BAMS
. . . . . . . . . . . . . CONVERT LONGITUDE TO BAMS
. . . . . . . . . . . . . IF LONGITUDE IS WEST
. . . . . . . . . . . . . . THEN
. . . . . . . . . . . . .   . SUBTRACT DEGREES FROM 360
. . . . . . . . . . . . . . ELSE
. . . . . . . . . . . . .   . LEAVE DEGREES ALONE
. . . . . . . . . . . . . ENDIF
. . . . . . . . . . . . . DECODE LONGITUDE
. . . . . . . . . . . . . IF WITHIN TOLERANCE
. . . . . . . . . . . . . . THEN
. . . . . . . . . . . . .   . NO ERROR
. . . . . . . . . . . . . . ELSE
. . . . . . . . . . . . .   . ERROR
. . . . . . . . . . . . . ENDIF
. . . . . . . . . . . . . SET TOLERANDE ERROR STATUS
. . . . . . . . . . . . . DECODE LATITUDE
. . . . . . . . . . . . . IF WITHIN TOLERANCE
. . . . . . . . . . . . . . THEN
. . . . . . . . . . . . .   . NO ERROR
. . . . . . . . . . . . . . ELSE
. . . . . . . . . . . . .   . ERROR
. . . . . . . . . . . . . ENDIF
. . . . . . . . . . . . . SET TOLERANDE ERROR STATUS
. . . . . . . . . . . . . *IDENT EQ 6
. . . . . . . . . . . . . DECODE ALTITUDE COMPENSATION
. . . . . . . . . . . . . IF SIGN BIT = 1
. . . . . . . . . . . . . . THEN
. . . . . . . . . . . . .   . MAKE VALUE NEGATIVE
. . . . . . . . . . . . . . ELSE
. . . . . . . . . . . . .   . LEAVE VALUE POSITIVE
. . . . . . . . . . . . . ENDIF
. . . . . . . . . . . . . *IDENT EQ 7
. . . . . . . . . . . . .   DECODE OPTION
. . . . . . . . . . . . ENDCASE
. . . . . . . . . . . . IF *IN* IS LESS THAN *LWA*
. . . . . . . . . . . .   THEN
. . . . . . . . . . . .     INCREMENT *IN*
. . . . . . . . . . . .   ELSE
```

```
. . . . . . . . . . . RESET *IN* TO *FWA*
. . . . . . . . . .ENDIF
. . . . . . . . ENDDO
. . . . . ENDIF
. . . .ENDIF
. . .ELSE
. . . .CONTINUE PROCESSING OTHER COMMANDS
. . ENDIF
. . IF CONTROL COMMAND
. . .THEN
. . . .PROCESS CONTROL COMMAND
. . . .DECODE WORD COUNT
. . . .IF *IN* LESS THAN *LWA*
. . . . THEN
. . . . . INCREMENT *IN*
. . . . ELSE
. . . . . RESET *IN* TO *FWA*
. . . .ENDIF
. . .ELSE
. . . .CONTINUE PROCESSING OTHER COMMANDS
. . ENDIF
. . IF COMMAND IS A MULTI-MESSAGE TRANSFER
. . .THEN
. . . .PROCESS MULTI-MESSAGE TRANSFER
. . .ELSE
. . . .CONTINUE PROCESSING OTHER COMMANDS
. . ENDIF
. . ALL OTHER COMMANDS ARE TREATED AS NO-OPS
ENDDO
OUTPUT PROCESSING STARTS HERE
IF SELF-TEST IS IN PROGRESS
. THEN
. . DECREMENT SELF-TEST COUNTER
. . IF SELF-TEST HAS NOT TIMED OUT
. . .THEN
. . . .SET FLAG TO SKIP OUTPUT THIS CYCLE
. . . .SET RECEIVE BUSY BIT
. . .ELSE
. . . .FORCE OUTPUT OF NON-ZERO BIT STATUS WORD
. . ENDIF
. ELSE
. . CONTINUE PROCESSING
ENDIF
IF CP AND PP DATA AVAILABLE BITS ARE THE SAME
. THEN
. . IF BC HAS NOT REQUESTED PREVIOUS DATA
. . .THEN
. . . .SET FLAG TO SKIP OUTPUT DATA THIS CYCLE
. . .ELSE
. . . .ALLOW OUTPUT DATA THIS CYCLE
. . ENDIF
. . IF INITIATE PROCESSING MODE/DISCRETE NOT ACTIVE
. . .THEN
. . . .SET FLAG TO SKIP OUTPUT DATA THIS CYCLE
. . .ELSE
. . . .ALLOW OUTPUT DATA THIS CYCLE
```

```
. . ENDIF
. . IF LOCKON DETECT NOT SET ( SYSTEM NOT READY )
. .  .THEN
. .  .  .SET FLAG TO SKIP OUTPUT THIS CYCLE
. .  .  .TURN SYSTEM READY BIT OFF
. .  .ELSE
. .  .  .TURN SYSTEM READY BIT ON
. . ENDIF
. . IF TRAIL NOT OUT
. .  .THEN
. .  .  .SET FLAG TO SKIP OUTPUT THIS CYCLE
. .  .  .TURN TRAIL BIT OFF
. .  .ELSE
. .  .  .TURN TRAIL BIT ON
. . ENDIF
. . IF MULTI-MESSAGE ERROR HAS OCCURRED
. .  .THEN
. .  .  .SET BIT 2 IN BIT STATUS WORD
. .  .ELSE
. .  .  .LEAVE BIT 2 OFF
. . ENDIF
. . IF BIT STATUS HAS CHANGED (BIT CHANGES LOW TO HIGH)
. .  .THEN
. .  .  .PLACE RT WITH T/F BIT SET AND NEW FAULTS
. .  .  .INTO OUTPUT BUFFER
. .  .ELSE
. .  .  .CHANGE OLD BIT STATUS TO NEW BIT STATUS
. . ENDIF
. . IF OUTPUT DATA SHOULD BE SENT
. .  .THEN
. .  .  .CASE MODE VALUE (OPTION)
. .  .  .GO TO ( 820, 840, 860, 880), OPTION+1
. .  .  .  .  *OPTION EQ 0
. .  .  .  .  PROCESS MAGNETOMETER DATA
. .  .  .  .  IF PRELIMINARY EVENT FLAG ON
. .  .  .  .  . THEN
. .  .  .  .  .  . SET RT DATA AVAILABLE
. .  .  .  .  .  . COMPUTE PRELIMINARY TIME LATE
. .  .  .  .  .  . RESET FLAGS TO ZERO
. .  .  .  .  .  . INSERT INTO OUTPUT BUFFER
. .  .  .  .  . ELSE
. .  .  .  .  .  . OMIT PRELIMINARY EVENT DATA
. .  .  .  .  . ENDIF
. .  .  .  .  . DOWHILE MAD EVENT POSSIBLE
. .  .  .  .  .  . IF CONFIRM DETECT FLAG ON
. .  .  .  .  .  .  .THEN
. .  .  .  .  .  .  .  .SET RT DATA AVAILABLE
. .  .  .  .  .  .  .  .COMPUTE FINAL TIME LATE
. .  .  .  .  .  .  .  .INSERT INTO OUTPUT BUFFER
. .  .  .  .  .  .  .  .INSERT SLANT RANGE INTO OUTPUT
. .  .  .  .  .  .  .  .RESET FLAG TO ZERO
. .  .  .  .  .  .  .ELSE
. .  .  .  .  .  .  .  .OMIT CONFIRM DETECT DATA
. .  .  .  .  .  . ENDIF
. .  .  .  .  . ENDDO
. .  .  .  .  *OPTION EQ 1
```

```
. . . . .  TRANSMIT DIGITAL MAGNETOMETER DATA
. . . . .  PROCESS AN/AYK-14 DATA
. . . . .  SET RT DATA AVAILABLE
. . . . .  SEND 2 WORDS OF TOTAL MAD FIELD
. . . . .  SEND ALTITUDE COMPENSATION
. . . . .  IF ALTITUDE COMPENSATION IS POSITIVE
. . . . .  . THEN
. . . . .  . . SET SIGN BIT TO 0
. . . . .  . ELSE
. . . . .  . . SET SIGN BIT TO 1
. . . . .  ENDIF
. . . . .  *OPTION EQ 2
. . . . .  TRANSMIT DIGITAL MAGNETOMETER DATA
. . . . .  SET RT DATA AVAILABLE
. . . . .  SEND 2 WORDS OF TOTAL MAD FIELD
. . . . .  SEND ALTITUDE COMPENSATION = ALTITUDE
. . . . .  *OPTION EQ 3
. . . . .  PROCESS AN/AYK-14 DATA
. . . . .  THIS OPTION IS TREATED AS A NO-OP
. . .  .ENDCASE
. . .  .SET WORD COUNT IN RT STATUS
. .  .ELSE
. . .  .OMIT ALL OUTPUT DATA THIS CYCLE
. .  ENDIF
. .  IF RT STATUS HAS CHANGED OR IF SOMETHING IS IN OUTPUT BUFFER
. .  .THEN
. . .  .PUT RT STATUS WORD INTO OUTPUT BUFFER
. . .  .IF STATUS SENT FLAG IS RESET
. . .  THEN
. . . . .  IF DATA SENT FLAG IS RESET
. . . . .  . THEN
. . . . . . . IF DATA IN OUTPUT BUFFER
. . . . . . . .  .THEN
. . . . . . . .  .SET DATA SENT FLAG
. . . . . . . .  .ELSE
. . . . . . . .  .DO NOT SET FLAG
. . . . . . .  ENDIF
. . . . . . .  PACK OUTPUT BUFFER
. . . . . . .  CONSTRUCT HEADER WORD
. . . . . . .  CHANGE OLD BIT STATUS TO NEW BIT STATUS
. . . . . . .  SET OP DATA AVAILABLE BIT
. . . . . . .  SET STATUS SENT FLAG
. . . . . .  ELSE
. . . . . .  . SET APPROPRIATE BIT IN ERROR WORD 8
. . . . .  ENDIF
. . . .  ELSE
. . . . .  SET APPROPRIATE BIT IN ERROR WORD 2
. . .  .ENDIF
. .  .ELSE
. . .  .DO NOT PUT RT INTO OUTPUT BUFFER
. .  ENDIF
.  ELSE
.  . SET APPROPRIATE BIT IN ERROR WORD 1
ENDIF
```

```
.   . VALUE - 0 OR 1
CODING HISTORY
.   . 1. PROGRAMMED--ALEX PODLECKI          10/26/77
END OF ABSTRACT
```

```
      ZERO OUT OLD VALUE
      MERGE IN NEW VALUE
```

```
      SUBROUTINE MSPPACK(N,SOURCE,LSB,RESULT)
      ABSTRACT
           THIS ROUTINE PACKS A REAL INTO A BINARY OF REQUESTED SIZE

           N - NUMBER OF BITS IN RESULT

           SOURCE - REAL VALUE TO BE PACKED

           LSB - REAL VALUE OF LEAST SIGNIFICANT BINARY BIT IN RESULT

           RESULT - BINARY OUTPUT WORD

      CODING HISTORY
           1. PROGRAMMED--ALEX PODLECKI        10/26/77

      END OF ABSTRACT
```

```
      CALCULATE VALUE OF MOST SIGNIFICANT BIT
      DO WHILE MORE BITS REQUIRED IN RESULT
      .  . SHIFT PREVIOUS RESULT OVER ONE BIT
      .  . DETERMINE IF NEXT BIT SHOULD BE SET
      .  . MERGE IN WITH PREVIOUS RESULT
      .  . CALCULATE REMAINDER FROM SOURCE VALUE
      .  . CALCULATE VALUE OF NEXT BIT
      ENDDO
```

```
      SUBROUTINE LOCATE(X,Y,ZLAT,ZLONG)
      ABSTRACT
           THIS ROUTINE COMPUTES CURRENT LATITUDE AND LONGITUDE
           USING DELTA X AND DELTA Y FROM GRP

           X - DELTA X (FEET)

           Y - DELTA Y (FEET)

           ZLAT - RESULTANT LATITUDE (DEGREES)

           ZLONG - RESULTANT LONGITUDE (DEGREES)

      CODING HISTORY
           1. STOLEN FROM ATOKEY -- ALEX PODLECKI  11/28/77

      END OF ABSTRACT
```

```
CONVERT GRP LATITUDE TO DEGREES
IF LATITUDE IS SOUTH
. THEN
.  . MAKE DEGREES NEGATIVE
. ELSE
.  . LEAVE SIGN AS POSITIVE
ENDIF
CONVERT GRP LONGITUDE TO DEGREES
IF LONGITUDE IS WEST
. THEN
.  . MAKE DEGREES NEGATIVE
. ELSE
.  . LEAVE SIGN AS POSITIVE
ENDIF
ADD DELTA X TO LATITUDE
ADD DELTA Y TO LONGITUDE
```

NAVIGATION INTERFACE UNIT MODULE

(NIU)

```
      PROGRAM NIUDRIV
      ABSTRACT
           THIS PROGRAM IS A DRIVER TEST PROGRAM FOR THE NIU1 MODULE.

      CODING HISTORY
           1. PROGRAMMED--ALEX PODLECKI        11/09/77

      END OF ABSTRACT
```

```
      ZERO INPUT BUFFER
      DOWHILE ANOTHER INPUT BUFFER WORD AVAILABLE
      .  . INSERT ZERO WORD
      ENDDO
      DEGREES TO RADIANS
      NO INITIAL FAULTS
      INITIALIZE CP,PP DATA AVAILABLE WORDS
      INITALIZE STATUS/DATA SENT FLAGS
      SELECT TACAN FOR PILOT/ATO
      RESPONSE TO NO COMMAND
      SELF-TEST SEQUENCE
      INSERT INITIATE SELF-TEST INTO INPUT BUFFER
      EXECUTE NIU
      DOWHILE NIU1 APPEARS TO BE BUSY
      .  . EXECUTE NIU1
      ENDDO
      INSERT TRANSMIT BIT STATUS INTO INPUT BUFFER
      EXECUTE NIU
      INITIALIZATION SEQUENCE
      INSERT INITIALIZE TERMINAL INTO INPUT  BUFFER
      INSERT TRANSMIT BIT STATUS INTO INPUT BUFFER
      INSERT INITIATE PROCESSING INTO INPUT BUFFER
      EXECUTE NIU
      DATA TRANSFER SEQUENCE
      NORMAL DATA TRANSFER OF 6 WORDS
      WORD 1 - TACTICAL RANGE = 1.0078125 MILES
      WORD 2 - TACTICAL BEARING = 180.010986
      WORD 3 - DRIFT ANGLE = 90.010986
      WORD 4 - PILOTS HEADING = 180.0
      WORD 5 - ATOS HEADING = 180.0
      WORD 6 - CONTROL DATA WORD
      INSERT TRANSMIT BIT STATUS COMMAND INTO BUFFER
      EXECUTE NIU1
      REQUEST DATA TRANSFER SEQUENCE
      INSERT REQUEST FOR DATA INTO INPUT BUFFER
      EXECUTE NIU
      EXECUTE NIU WITH NULL INPUT 5 TIMES
      ERROR PROCESSING SECTION
      SET DOPPLER FAULT BIT
      EXECUTE NIU1
      END OF PROGRAM
```

```
      SUBROUTINE XNIU(IN,LWA)
      ABSTRACT
           THIS ROUTINE PERFORMS 3 FUNCTIONS
                1. IT PRINTS THE CONTENTS OF THE CURRENT INPUT BUFFER
                2. IT EXECUTES THE NIU1 MODULE
                3. IT PRINTS THE CONTENTS OF THE RESULTANT OUTPUT
                   BUFFER AND RESETS THE PP BIT TO SIGNIFY ACCEPTANCE

           IN - POINTER TO FIRST WORD OF INPUT

           LWA - POINTER TO LAST WORD ADDRESS OF INPUT BUFFER

      CODING HISTORY
           1. PROGRAMMED--ALEX PODLECKI        11/11/77

      END OF ABSTRACT
```

```
      PRINT INPUT BUFFER
      DOWHILE SOMETHING IN INPUT BUFFER
      .  . EXPAND INPUT WORD
      .  . PRINT INPUT WORD BIT-BY-BIT
      ENDDO
      IF EMPTY INPUT BUFFER AND PRINT MESSAGES SELECTED
      . THEN
      .  . PRINT INFORMATIVE MESSAGE
      . ELSE
      .  . OMIT MESSAGE
      ENDIF
      EXECUTE THE NIU1 MODULE
      PRINT OUTPUT BUFFER
      IF OUTPUT BUFFER EMPTY
      . THEN
      .  . IF PRINT MESSAGES SELECTED
      .  .  .THEN
      .  .  .  .PRINT INFORMATIVE MESSAGE
      .  .  .ELSE
      .  .  .  .OMIT MESSAGE
      .  . ENDIF
      . ELSE
      .  . PRINT HEADER WORD
      .  . DOWHILE SOMETHING IN THE OUTPUT BUFFER
      .  .  .  .EXPAND OUTPUT WORD
      .  .  .  .PRINT OUTPUT WORD BIT-BY-BIT
      .  . ENDDO
      .  . TOGGLE PP DATA AVAILABLE BIT
      ENDIF
      END OF ROUTINE
```

```
      SUBROUTINE NIU1
      ABSTRACT
           THIS ROUTINE UPDATES FTP BEARING FOR HELOINP
           AND TRANSFERS AN/AYK-14 NAVIGATION DATA TO DTOA.

      CODING HISTORY
           1. PROGRAMMED--ALEX PODLECKI        10/28/77

      END OF ABSTRACT
```

```
. . . . . . THEN
. . . . . .  . INCREMENT *IN*
. . . . . . ELSE
. . . . . .  . SET *IN* TO *FWA*
. . . . . ENDIF
. . . . . TRANSFER PILOTS HEADING TO DTOA
. . . . . IF *IN* LESS THAN *LWA*
. . . . . . THEN
. . . . . .  . INCREMENT *IN*
. . . . . . ELSE
. . . . . .  . SET *IN* TO *FWA*
. . . . . ENDIF
. . . . . TRANSFER ATO HEADING TO DTOA
. . . . . IF *IN* LESS THAN *LWA*
. . . . . . THEN
. . . . . .  . INCREMENT *IN*
. . . . . . ELSE
. . . . . .  . SET *IN* TO *FWA*
. . . . . ENDIF
. . . . . IGNORE CONTROL DATA WORD
. . . . . IF *IN* LESS THAN *LWA*
. . . . . . THEN
. . . . . .  . INCREMENT *IN*
. . . . . . ELSE
. . . . . .  . SET *IN* TO *FWA*
. . . . . ENDIF
. . . . .ENDIF
. . ENDCASE
ENDDO
OUTPUT PROCESSING
. THEN
. . DECREMENT TIME LEFT IN SELF-TEST
. . IF SELF-TEST HAS  TIMED OUT
. .  .THEN
. .  .  .SETUP OUTPUT OF RT AND BIT STATUS WORDS
. .  .ELSE
. . ENDIF
. ELSE
. . IF BIT STATUS HAS CHANGED
. .  .THEN
. .  .  .PLACE RT WITH T/F BIT SET AND NEW FAULTS FER
. .  .  .INTO OUTPUT BUFFER
. .  .ELSE
. .  .  .SET OLD BIT TO NEW BIT
. . ENDIF
. . IF OUTPUT DATA SHOULD BE SENT
. .  .THEN
. .  .  .IF TIME FOR 1 HZ DATA
. .  .  .  THEN
. . . . . SET RT DATA AVAILABLE BIT
. . . . . SET RT WORD COUNT
. . . . . PLACE EQUIPMENT STATUS WORDS INTO
. . . . . OUTPUT BUFFER
. . . . . IF TACAN SELECTED
. . . . . . THEN
. . . . . .  . SET TACAN BITS IN STATUS WORD 2
```

```
                . ELSE
                .   . SET COMPUTER SELECTED BITS
                . ENDIF
              . ELSE
              .   . NO OUTPUT THIS CYCLE
              .ENDIF
            . ENDIF
          ENDIF
          IF PREVIOUS OUTPUT READ BY PP
          . THEN
          .   . IF AN/AYK-14 HAS NOT REQUESTED PREVIOUS DATA
          .   .THEN
          .   .   .OUTPUT THIS CYCLE
          .   .   .CHECK FOR RT STATUS CHANGE
          .   .   .IF RT STATUS HAS CHANGED
          .   .   .   THEN
          .   .   .   .  PUT RT INTO OUTPUT BUFFER
          .   .   .   ELSE
          .   .   .   .  LEAVE OUTPUT ALONE
          .   .   .   .ENDIF
          .   .   .   .SEND DATA IN BUFFER TO IIU
          .   .   .   .IF SOMETHING IN OUTPUT BUFFER
          .   .   .   .  THEN
          .   .   .   .  .  PACK BUFFER FOR PP
          .   .   .   .  .  CONSTRUCT HEADER WORD
          .   .   .   .  .  RESET CP DATA AVAILABLE BIT
          .   .   .   .  .  IF DATA HAS BEEN SENT
          .   .   .   .  .  . THEN
          .   .   .   .  .  .   . SET DATA SENT FLAG
          .   .   .   .  .  . ELSE
          .   .   .   .  .  .   . FLAG NOT SET
          .   .   .   .  .  . ENDIF
          .   .   .   .  .  ELSE
          .   .   .   .  .  .  NO DATA TO BE OUTPUT
          .   .   .   .  .ENDIF
          .   .   .  .ELSE
          .   .   .  .SET ERROR WORD WHEN STATUS OR DATA SENT FLAG NOT RESET
          .   . ENDIF
          . ELSE
          .   . SET ERROR WORD 1 - DATA NOT ACCEPTED BY PP
          ENDIF
          END OF MODULE
```

```
      SUBROUTINE ADVANCE(POINTER,LWA)
      ABSTRACT
           THIS ROUTINE INCREMENTS A POINTER BY 1. IF THE POINTER WAS
           ALREADY SET TO AN LWA, THE POINTER IS RESET TO 1.

           POINTER - CURRENT VALUE OF POINTER

           LWA - LAST WORD ADDRESS FOR POINTER

      CODING HISTORY
           1. PROGRAMMED--ALEX PODLECKI        11/04/77

      END OF ABSTRACT
```

```
IF POINTER IS LESS THAN LWA
. THEN
.  . INCREMENT POINTER
. ELSE
.  . SET POINTER TO FWA
ENDIF
```

```
      SUBROUTINE EXPAND(N,IN,OUT)
      ABSTRACT
           THIS ROUTINE EXPANDS A WORD INTO AN N WORD ARRAY SUCH THAT
           WORD 1 CONTAINS BIT N-1, WORD 2 CONTAINS BIT N-2, ...,
           AND WORD N CONTAINS BIT 0 ( RIGHT JUSTIFIED WITH ZERO FILL )

           N - NUMBER OF BITS TO BE EXPANDED

           IN - INPUT WORD TO BE EXPANDED

           OUT - OUTPUT ARRAY TO RECEIVE EXPANSION

      CODING HISTORY
           1. PROGRAMMED--ALEX PODLECKI        11/07/77

      END OF ABSTRACT
```

```
DOWHILE ANOTHER BIT TO BE EXPANDED
.  . MASK OUT DESIRED BIT
.  . SET UP FOR NEXT BIT
ENDDO
```

```
      SUBROUTINE PACKPP( NRT, N)
      ABSTRACT
           THIS ROUTINE IS A DUMMY SUBSTITUTE FOR THE ACTUAL PACKPP.

           NRT - NUMBER CORRESPONDING TO RT

           N - NUMBER OF WORDS TO BE *PACKED*

      CODING HISTORY
           1. PROGRAMMED--ALEX PODLECKI        11/07/77

      END OF ABSTRACT
```

EXIT

ORDNANCE LAUNCH CONTROL SET MODULE

(OLCS)

```
      PROGRAM DRIVER
      ABSTRACT
           DRIVER PROGRAM TO TEST OLCS SUBROUTINE.
      CODING HISTORY
           1. PROGRAMMED  J. MANGES  CSC   DEC,JAN 1977,78
      END OF ABSTRACT
```

SET INITIAL CHUTE LOADING

```
CHECK INITIALIZATION
CHECK ACTION OF ERROR WORDS
ERROR WORD 2
SET STATUS SENT FLAG TO 1
CHECK VARIABLES, CALL OLCS, AND CHECK THE RESULTS
ERROR WORD 3
SENT DATA SENT FLAG TO 1
CHECK VARIABLES, CALL OLCS, AND CHECK THE RESULTS
ERROR WORD 1
SET DATA AVAILABLE FLAGS TO OPPOSITE VALUES
CHECK VARIABLES, CALL OLCS, AND CHECK THE RESULTS
RESET ERROR WORDS AND FLAGS
CHANGE IN BIT STATUS WORD
SET A BIT IN THE BIT STATUS WORD
CHECK VARIABLES, CALL OLCS, AND CHECK THE RESULTS
RESET FLAGS
OSRU SELF TEST SEQUENCE
SET COMMAND WORD INTO THE INPUT BUFFER
SET BUFFER POINTER
CHECK VARIABLES, CALL OLCS, AND CHECK THE RESULTS
RESET FLAGS
DECREMENT THE BIT COUNTER
DO WHILE I IS LESS THAN 372
ENDDO
DO WHILE I IS LESS THAN 3
ENDDO
OSRU INITIALIZATION SEQUENCE
SET INITIALIZE TERMINAL COMMAND IN INPUT BUFFER
SET BUFFER POINTER
SET INITIATE PROCESSING COMMAND IN INPUT BUFFER
SET BUFFER POINTER
RESET FLAGS
OSRU NORMAL DATA TRANSFER SEQUENCE
SET NORMAL DATA TRANSFER COMMAND IN INPUT BUFFER
SET BUFFER POINTER
SET DATA SENT FLAG
OSRU DISCRETES
TORPEDO MANUAL LAUNCH COMMAND
SET MASTER ARM ON
SET TORPEDO ARM ON
SET MANUAL TORPEDO FIRE DISCRETE
CHECK VARIABLES, CALL OLCS, AND CHECK THE RESULTS
RESET FLAGS
TORPEDO AWAY SIGNAL
DO WHILE I IS LESS THAN SEVEN
ENDDO
RESET DISCRETE
RESET FLAGS
ATTEMPT TO FIRE TORPEDO WITH MASTER ARM OFF
SET MASTER ARM OFF
SET MANUAL TORPEDO FIRE DISCRETE
CHECK VARIABLES, CALL OLCS, AND CHECK THE RESULTS
RESET DISCRETES
ATTEMPT TO FIRE TORPEDO WITH TORPEDO ARM OFF
SET TORPEDO ARM OFF
SET MANUAL TORPEDO FIRE DISCRETE
```

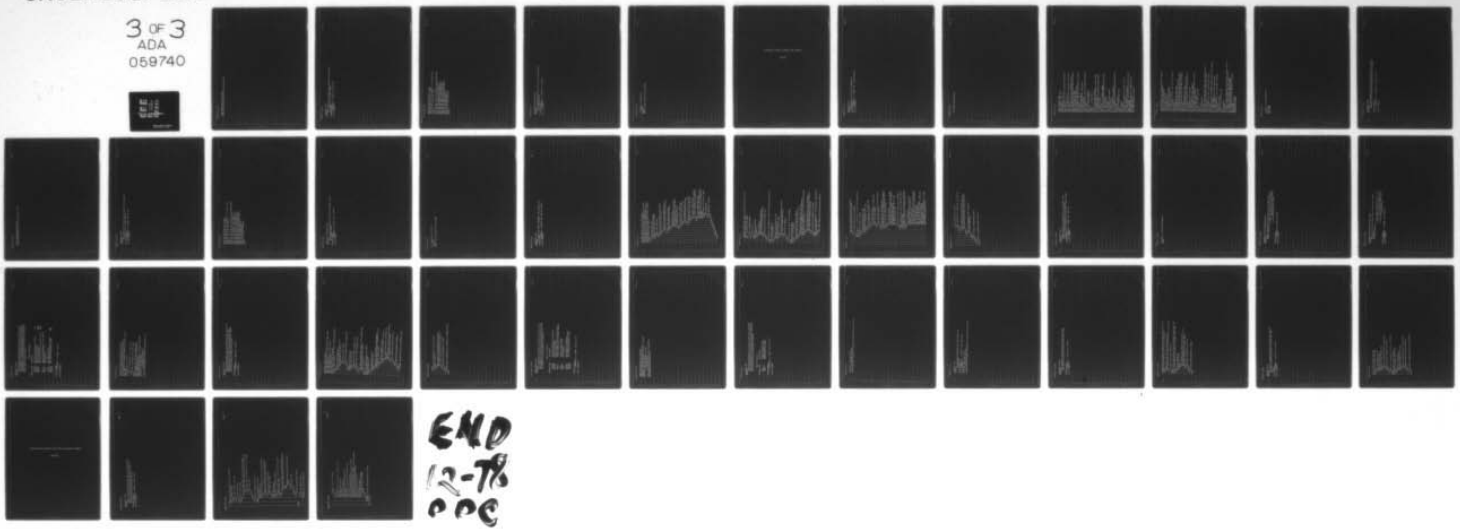
```
      CHECK VARIABLES, CALL OLCS,AND CHECK THE RESULTS
      ATTEMPT TO FIRE THREE TORPEDOES
      SET ARM AND FIRE DISCRETES
      FIRE THE SECOND TORPEDO
      RESET TORPEDO FIRE
      CALL SETABIT(IATOTOG(1),2,0)
      ATTEMPT TO FIRE A THIRD TORPEDO
      SET TORPEDO FIRE DISCRETE
      CHECK VARIABLES, CALL OLCS, AND CHECK THE RESULTS
      RESET DISCRETES
      RESET FLAGS
      SONOBUOY SELECT AND LAUNCH MODE
      SET SONOBUOY AUTO SELECT AND LAUNCH MODE
      CHECK VARIABLES, CALL OLCS, AND CHECK THE RESULTS
      SET SONOBUOY MANUAL LAUNCH AND SELECT DISCRETE
      CHECK VARIABLES, CALL OLCS, AND CHECK THE RESULTS
      RESET FLAGS
      MANUAL SONOBUOY SELECT CHUTE 19
      SET MANUAL SONOBUOY SELECT CHUTE 19 DISCRETE
      CHECK VARIABLES, CALL OLCS, AND CHECK THE RESULTS
      RESET FLAGS
      SONOBUOY MANUAL LAUNCH COMMAND
      SET MASTER ARM ON
      SET MANUAL SONOBUOY LAUNCH DISCRETE
      CHECK VARIABLES, CALL OLCS, AND CHECK THE RESULTS
      RESET SONOBUOY LAUNCH COMMAND
      SONOBUOY AWAY SIGNAL
      DO WHILE I IS LESS THAN SEVEN
      ENDDO
      RESET FLAGS
      RESET ALL DISCRETES
      RESET FLAGS
      SEQUENCE 1- AUTO SELECT MODE, AUTO LAUNCH COMMAND (CHUTE 1)
      SET SONOBUOY AUTO SELECT AND LAUNCH MODE DISCRETE
      SET MASTER ARM ON
      CHECK VARIABLES, CALL OLCS, AND CHECK THE RESULTS
      RESET FLAGS
      SET CONTROL COMMAND DATA TRANSFER COMMAND WORD IN INPUT BUFFER
      SET BUFFER POINTER
      SET CONTROL COMMAND DATA WORD IN INPUT BUFFER
      SET BUFFER POINTER
      CHECK VARIABLES, CALL OLCS, AND CHECK THE RESULTS
      RESET FLAGS
      SONOBUOY AWAY SIGNAL
      DO WHILE I IS LESS THAN SEVEN
      ENDDO
      RESET DISCRETES
      RESET FLAGS
      SEQUENCE 2- MANUAL SELECT MODE, MANUAL LAUNCH COMMAND (CHUTE 16)
      SET SONOBUOY MANUAL SELECT AND LAUNCH MODE DISCRETE
      SET SONOBUOY MANUAL SELECT CHUTE 16 DISCRETE
      SET SONOBUOY MANUAL LAUNCH COMMAND DISCRETE
      CHECK VARIABLES, CALL OLCS, AND CHECK THE RESULTS
      RESET FLAGS
      RESET LAUNCH COMMAND
      SONOBUOY AWAY SIGNAL
```

```
      DO WHILE I IS LESS THAN SEVEN
      ENDDO
      CALL PRINT
      CALL OLCS
```

```
      SUBROUTINE FLAGS
      ABSTRACT
           THIS ROUTINE RESETS THE VALUES OF THE BUFFER AND DATA AVAILAB
           FLAGS TO A STATE IN WHICH THE PACKPP ROUTINE MAY BE CALLED.

      CODING HISTORY
           1.  PROGRAMMED   J. MANGES CSC  APRIL 11, 1978

      END OF ABSTRACT
```

AD-A059 740 COMPUTER SCIENCES CORP HUNTINGDON VALLEY PA F/G 15/1
LAMPS SEAS SIMULATION SOFTWARE SUPPORT.(U)
JUN 78 N62269-75-C-0001
UNCLASSIFIED NL

3 OF 3
ADA
059740



END
12-78
DDC

```
RESET OUTPUT BUFFER FULL FLAGS
EQUATE DATA AVAILABLE AND PP DATA AVAILABLE FLAGS
```

```
      SUBROUTINE PRINT
      ABSTRACT
           PRINT DISPLAYS THE VALUES OF ALL RELEVANT OLCS VARIABLES
           WHEN CALLED BY THE DRIVER PROGRAM
      CODING HISTORY
           1. PROGRAMMED  J. MANGES  CSC  DEC 1977
      END OF ABSTRACT
```

```
      WRITE OUT THE VALUES OF THE FLAGS
      WRITE OUT BIT BY BIT THE RT WORDS
      WRITE OUT BIT BY BIT THE CONTENTS OF THE INPUT BUFFER
      DO WHILE I IS LESS THAN FIFTEEN
      ENDDO
      WRITE OUT BIT BY BIT THE VALUES OF THE DISCRETE ARRAY
      WRITE OUT THE VALUES OF THE BUOYIC ARRAY
      WRITE OUT THE SONOBUOY SPLASH POINTS
      WRITE OUT THE SONOBUOY WATER ENTRY TIMES
      WRITE OUT THE VALUE OF THE SELECTED BUOY
      WRITE OUT THE VALUE OF THE BIT COUNTER
      WRITE OUT THE TORPEDO AWAY SIGNAL COUNTERS
      WRITE OUT THE BUFFER POINTERS
      WRITE OUT THE ERROR WORDS
      DO WHILE I IS LESS THAN THREE
      ENDDO
```

```
      SUBROUTINE BITS(JVALUE,NUM)
      ABSTRACT
           THIS SUBROUTINE PRINTS OUT BIT BY BIT THE FIRST 16 BITS
           OF THE WORD JVALUE.
      CODING HISTORY
           1.  PROGRAMMED  J. MANGES  CSC  12/28/77
      END OF ABSTRACT
```

```
      DO WHILE I IS LESS THAN 16
      .  . ENDIF
      ENDDO
      WRITE OUT THE VALUES OF THE BIT ARRAY
```

ORDNANCE LAUNCH CONTROL SET MODULE

(OLCS)

```
      PROGRAM DRIVER
      ABSTRACT
           DRIVER PROGRAM TO TEST OLCS SUBROUTINE.
      CODING HISTORY
           1. PROGRAMMED  J. MANGES  CSC   DEC,JAN 1977,78
      END OF ABSTRACT
```

SET INITIAL CHUTE LOADING

```
CHECK INITIALIZATION
CHECK ACTION OF ERROR WORDS
ERROR WORD 2
SET STATUS SENT FLAG TO 1
CHECK VARIABLES, CALL OLCS, AND CHECK THE RESULTS
ERROR WORD 3
SENT DATA SENT FLAG TO 1
CHECK VARIABLES, CALL OLCS, AND CHECK THE RESULTS
ERROR WORD 1
SET DATA AVAILABLE FLAGS TO OPPOSITE VALUES
CHECK VARIABLES, CALL OLCS, AND CHECK THE RESULTS
RESET ERROR WORDS AND FLAGS
CHANGE IN BIT STATUS WORD
SET A BIT IN THE BIT STATUS WORD
CHECK VARIABLES, CALL OLCS, AND CHECK THE RESULTS
RESET FLAGS
OSRU SELF TEST SEQUENCE
SET COMMAND WORD INTO THE INPUT BUFFER
SET BUFFER POINTER
CHECK VARIABLES, CALL OLCS, AND CHECK THE RESULTS
RESET FLAGS
DECREMENT THE BIT COUNTER
DO WHILE I IS LESS THAN 372
ENDDO
DO WHILE I IS LESS THAN 3
ENDDO
OSRU INITIALIZATION SEQUENCE
SET INITIALIZE TERMINAL COMMAND IN INPUT BUFFER
SET BUFFER POINTER
SET INITIATE PROCESSING COMMAND IN INPUT BUFFER
SET BUFFER POINTER
RESET FLAGS
OSRU NORMAL DATA TRANSFER SEQUENCE
SET NORMAL DATA TRANSFER COMMAND IN INPUT BUFFER
SET BUFFER POINTER
SET DATA SENT FLAG
OSRU DISCRETES
TORPEDO MANUAL LAUNCH COMMAND
SET MASTER ARM ON
SET TORPEDO ARM ON
SET MANUAL TORPEDO FIRE DISCRETE
CHECK VARIABLES, CALL OLCS, AND CHECK THE RESULTS
RESET FLAGS
TORPEDO AWAY SIGNAL
DO WHILE I IS LESS THAN SEVEN
ENDDO
RESET DISCRETE
RESET FLAGS
ATTEMPT TO FIRE TORPEDO WITH MASTER ARM OFF
SET MASTER ARM OFF
SET MANUAL TORPEDO FIRE DISCRETE
CHECK VARIABLES, CALL OLCS, AND CHECK THE RESULTS
RESET DISCRETES
ATTEMPT TO FIRE TORPEDO WITH TORPEDO ARM OFF
SET TORPEDO ARM OFF
SET MANUAL TORPEDO FIRE DISCRETE
```

```
      CHECK VARIABLES, CALL OLCS,AND CHECK THE RESULTS
      ATTEMPT TO FIRE THREE TORPEDOES
      SET ARM AND FIRE DISCRETES
      FIRE THE SECOND TORPEDO
      RESET TORPEDO FIRE
      CALL SETABIT(IATOTOG(1),2,0)
      ATTEMPT TO FIRE A THIRD TORPEDO
      SET TORPEDO FIRE DISCRETE
      CHECK VARIABLES, CALL OLCS, AND CHECK THE RESULTS
      RESET DISCRETES
      RESET FLAGS
      SONOBUOY SELECT AND LAUNCH MODE
      SET SONOBUOY AUTO SELECT AND LAUNCH MODE
      CHECK VARIABLES, CALL OLCS, AND CHECK THE RESULTS
      SET SONOBUOY MANUAL LAUNCH AND SELECT DISCRETE
      CHECK VARIABLES, CALL OLCS, AND CHECK THE RESULTS
      RESET FLAGS
      MANUAL SONOBUOY SELECT CHUTE 19
      SET MANUAL SONOBUOY SELECT CHUTE 19 DISCRETE
      CHECK VARIABLES, CALL OLCS, AND CHECK THE RESULTS
      RESET FLAGS
      SONOBUOY MANUAL LAUNCH COMMAND
      SET MASTER ARM ON
      SET MANUAL SONOBUOY LAUNCH DISCRETE
      CHECK VARIABLES, CALL OLCS, AND CHECK THE RESULTS
      RESET SONOBUOY LAUNCH COMMAND
      SONOBUOY AWAY SIGNAL
      DO WHILE I IS LESS THAN SEVEN
      ENDDO
      RESET FLAGS
      RESET ALL DISCRETES
      RESET FLAGS
      SEQUENCE 1- AUTO SELECT MODE, AUTO LAUNCH COMMAND (CHUTE 1)
      SET SONOBUOY AUTO SELECT AND LAUNCH MODE DISCRETE
      SET MASTER ARM ON
      CHECK VARIABLES, CALL OLCS, AND CHECK THE RESULTS
      RESET FLAGS
      SET CONTROL COMMAND DATA TRANSFER COMMAND WORD IN INPUT BUFFER
      SET BUFFER POINTER
      SET CONTROL COMMAND DATA WORD IN INPUT BUFFER
      SET BUFFER POINTER
      CHECK VARIABLES, CALL OLCS, AND CHECK THE RESULTS
      RESET FLAGS
      SONOBUOY AWAY SIGNAL
      DO WHILE I IS LESS THAN SEVEN
      ENDDO
      RESET DISCRETES
      RESET FLAGS
      SEQUENCE 2- MANUAL SELECT MODE, MANUAL LAUNCH COMMAND (CHUTE 16)
      SET SONOBUOY MANUAL SELECT AND LAUNCH MODE DISCRETE
      SET SONOBUOY MANUAL SELECT CHUTE 16 DISCRETE
      SET SONOBUOY MANUAL LAUNCH COMMAND DISCRETE
      CHECK VARIABLES, CALL OLCS, AND CHECK THE RESULTS
      RESET FLAGS
      RESET LAUNCH COMMAND
      SONOBUOY AWAY SIGNAL
```

```
DO WHILE I IS LESS THAN SEVEN
ENDDO
CALL PRINT
CALL OLCS
```

```
SUBROUTINE FLAGS
ABSTRACT
     THIS ROUTINE RESETS THE VALUES OF THE BUFFER AND DATA AVAILAB
     FLAGS TO A STATE IN WHICH THE PACKPP ROUTINE MAY BE CALLED.

CODING HISTORY
     1.  PROGRAMMED   J. MANGES CSC  APRIL 11, 1978

END OF ABSTRACT
```

```
RESET OUTPUT BUFFER FULL FLAGS
EQUATE DATA AVAILABLE AND PP DATA AVAILABLE FLAGS
```

```
      SUBROUTINE PRINT
      ABSTRACT
           PRINT DISPLAYS THE VALUES OF ALL RELEVANT OLCS VARIABLES
           WHEN CALLED BY THE DRIVER PROGRAM
      CODING HISTORY
           1. PROGRAMMED  J. MANGES  CSC  DEC 1977
      END OF ABSTRACT
```

```
      WRITE OUT THE VALUES OF THE FLAGS
      WRITE OUT BIT BY BIT THE RT WORDS
      WRITE OUT BIT BY BIT THE CONTENTS OF THE INPUT BUFFER
      DO WHILE I IS LESS THAN FIFTEEN
      ENDDO
      WRITE OUT BIT BY BIT THE VALUES OF THE DISCRETE ARRAY
      WRITE OUT THE VALUES OF THE BUOYIC ARRAY
      WRITE OUT THE SONOBUOY SPLASH POINTS
      WRITE OUT THE SONOBUOY WATER ENTRY TIMES
      WRITE OUT THE VALUE OF THE SELECTED BUOY
      WRITE OUT THE VALUE OF THE BIT COUNTER
      WRITE OUT THE TORPEDO AWAY SIGNAL COUNTERS
      WRITE OUT THE BUFFER POINTERS
      WRITE OUT THE ERROR WORDS
      DO WHILE I IS LESS THAN THREE
      ENDDO
```

```
      SUBROUTINE BITS(JVALUE,NUM)
      ABSTRACT
           THIS SUBROUTINE PRINTS OUT BIT BY BIT THE FIRST 16 BITS
           OF THE WORD JVALUE.
      CODING HISTORY
           1.  PROGRAMMED  J. MANGES  CSC  12/28/77
      END OF ABSTRACT
```

```
      DO WHILE I IS LESS THAN 16
      .   . ENDIF
      ENDDO
      WRITE OUT THE VALUES OF THE BIT ARRAY
```

```
SUBROUTINE OLCS
ABSTRACT
     SOFTWARE SIMULATION OF ORDNANCE LAUNCH CONTROL SET.
CODING HISTORY
     1. PROGRAMMED  J. MANGES  CSC  NOV-FEB, 1977-78
END OF ABSTRACT
```

```
      DESIGNATE UPPER LIMIT FOR BIT COUNTER
      DESIGNATE NUMBER OF WORDS IN CIRCULAR INPUT BUFFER
      DESIGNATE OLCS BIT IN IDAW(1) AND IDAW(2)
      INPUT PROCESSING- READ FROM CIRCULAR INPUT BUFFER
      LOCALIZE THE PRESENT VALUES OF THE BUFFER POINTERS
      IF INPUT BUFFER IS NOT EMPTY
      . THEN
      .   . CLEAR STATUS SENT FLAG
      .   . CHECK FOR NORMAL DATA TRANSFER
      .   . IF IRTBUFF(IB,7) EQUALS 1460628
      .   .  .THEN
      .   .  .  .RESET DATA SENT FLAG
      .   .  .ELSE
      .   .  .  .CHECK FOR CONTROL COMMAND DATA TRANSFER
      .   .  .  .IF IRTBUFF(IB,7) EQUALS 1442413
      .   .  .  .  THEN
      .   .  .  .  .  SET THE CONTROL COMMAND DATA WORD
      .   .  .  .  .  PROCESS THE COTROL COMMAND DATA WORDS
      .   .  .  .  ELSE
      .   .  .  .  .  CHECK FOR INITIALIZE TERMINAL
      .   .  .  .  .  IF IRTBUFF(IB,7) EQUALS 1440018
      .   .  .  .  .  . THEN
      .   .  .  .  .  .  . OVERRIDE BIT IF ON
      .   .  .  .  .  .  . RESET STATUS AND DATA SENT FLAGS
      .   .  .  .  .  .  . SET PP AND RT DATA AVBL BITS EQUAL
      .   .  .  .  .  .  . SET RECEIVE BUSY FLAG TO 0
      .   .  .  .  .  .  . SET QUIESCENT STATE FLAG
      .   .  .  .  .  . ELSE
      .   .  .  .  .  .  . CHECK FOR INITIATE PROCESSING
      .   .  .  .  .  .  . IF IRTBUFF(IB,7) EQUALS 1440048
      .   .  .  .  .  .  .  .THEN
      .   .  .  .  .  .  .  .  .RESET QUIESCENT STATE FLAG
      .   .  .  .  .  .  .  .  .SET FLAG TO TRANSMIT DATA WORDS
      .   .  .  .  .  .  .  .ELSE
      .   .  .  .  .  .  .  .  .CHECK FOR INITIATE BIT SELF TEST
      .   .  .  .  .  .  .  .  .IF IRTBUFF(IB,7) EQUALS 1440038
      .   .  .  .  .  .  .  .  .  THEN
      .   .  .  .  .  .  .  .  .  .  INITIALIZE COUNTER FOR BIT
      .   .  .  .  .  .  .  .  .  .  SET RECIEVE BUSY BIT IN RT STATUS WO
      .   .  .  .  .  .  .  .  .  .  SET FLAG TO TRANSMIT RT STATUS WORD
      .   .  .  .  .  .  .  .  .  .  ELSE
      .   .  .  .  .  .  .  .  .  .  .  CHECK FOR INVALID OLCS WORD
      .   .  .  .  .  .  .  .  .  .  .  SET ERROR FLAG IF IRTBUFF(IB,7) IS 0
      .   .  .  .  .  .  .  .  .  .  .  IDENTICALLY ZERO OR IS A CONTROL COM
      .   .  .  .  .  .  .  .  .  .  .  DATA WORD
      .   .  .  .  .  .  .  .  .  .  .  IF IRTBUFF(IB,7) NOT EQUAL TO ZERO
      .   .  .  .  .  .  .  .  .  .  .  . THEN
      .   .  .  .  .  .  .  .  .  .  .  .  . SET ERROR FLAG
      .   .  .  .  .  .  .  .  .  .  .  . ELSE
      .   .  .  .  .  .  .  .  .  .  .  .  . IRTBUFF(IB,7) IS ZERO
      .   .  .  .  .  .  .  .  .  .  .  ENDIF
      .   .  .  .  .  .  .  .  .  .  .ENDIF
      .   .  .  .  .  .  .  . ENDIF
      .   .  .  .  .  .  ENDIF
      .   .  .  .ENDIF
      .   . ENDIF
```

```
. . INCREMENT POINTER AND ZERO OUT THE BUFFER WORD JUST READ
. . CHECK FOR BUFFER WRAP AROUND
. . IF IB IS GREATER THAN NUMWDS
. . THEN
. . . .RESET POINTER POSITION
. . ELSE
. . . .CHECK FOR EOI
. . ENDIF
. . CHECK FOR EOI IN BUFFER
. . IF IB EQUALS IF
. . .THEN
. . . .RESET NWOUTRT(7)
. . . .BUFFER PROCESSING IS COMPLETED
. . . .JUMP TO THE BIT PROCESSING SECTION
. . .ELSE
. . . .CONTINUE TO LOOK AT INPUT BUFFER
. . ENDIF
. ELSE
. . NOTHING IN INPUT BUFFER SO SKIP INPUT PROCESSING THIS CYCLE
ENDIF
DO BIT COUNTER PROCESSING
IF BIT IS IN PROGRESS
. THEN
. . DECREASE THE BIT COUNTER BY ONE
. . CHECK FOR END OF BIT
. . IF NCOUNTR IS EQUAL TO ZERO
. . .THEN
. . . .BIT HAS ENDED SO SET RECEIVE BUSY BIT TO 0
. . . .SET FLAG TO TRANSMIT DATA WORDS
. . .ELSE
. . . .BIT IS ON SO CONTINUE
. . ENDIF
. ELSE
. . CONTINUE ON AS BIT IS NOT IN PROGRESS
ENDIF
CHECK FOR QUIESCENT STATE
IF OLCS NOT IN A QUIESCENT
. THEN
. . CHECK FOR BIT SELF TEST IN PROGRESS
. . IF BIT IS NOT ON OR HAS ONLY JUST BEGUN
. . .THEN
. . . .UPDATE ANY MANUAL INPUTS FROM THE OASP
. . . .DO SONOBUOY AND TORPEDO AWAY SIGNAL PROCESSING
. . . .CHECK TO SEE IF NECESSARY TO MAKE SPLASH OR WET CACULATI
. . . .IF SBCALC NOT EQUAL TO ZERO
. . . . . THEN
. . . . . MAKE SPLASH POINT CALCULATIONS FOR SONOBUOYS
. . . . . MAKE WATER ENTRY TIME CALCULATIONS FOR SONOBUOYS
. . . . . RESET CALCULATION-NECESSARY INDICATOR
. . . . . ELSE
. . . . . NO CALCULATION NECESSARY SO CONTINUE
. . . .ENDIF
. . . .CHECK TO SEE IF NECESSARY TO MAKE TSPLASH OR WRP CALCULA
. . . .IF TCALC EQUALS ONE
. . . . . THEN
. . . . . MAKE WATER ENTRY TIME CLACULATION FOR TORPEDOS
```

```
. . . . .  MAKE SPLASH POINT CALCULATION FOR TOPPEDO
. . . . .  RESET CALCULATION NECESSARY INDICATOR
. . . . ELSE
. . . . .  NO TORPEDO CALCULATIONS NECESSARY
. . . .ENDIF
. . . .CHECK TO SEE IF PP DATA AVBL FLAG AND DATA AVAILABLE FLA
. . . .ARE EQUAL
. . . .IF IPPDATA IS EQUAL TO IDATAVB
. . . .  THEN
. . . . .  INITIALIZE WORD COUNTER TO ONE
. . . . .  CHECK OUTPUT BUFFER FULL FLAGS
. . . . .  IF IBFUL1(7) AND IBFUL2(7) ARE ZERO
. . . . . . THEN
. . . . . .  . CHECK FOR CHANGE IN BIT STATUS WORD
. . . . . .  . IF IUPS IS NOT EQUAL TO ZERO
. . . . . .  . .THEN
. . . . . .  . .  .LOAD THE INPUT ARRAY
. . . . . .  . .  .SET THE T/F BIT IN THE RT STATUS WORD
. . . . . .  . .  .RESET THE WORD COUNTER
. . . . . .  . .ELSE
. . . . . .  . .  .NO 0 TO 1 TRANSITIONS IN BIT STATUS WORD
. . . . . .  . .  .RESET VALUE OF IOLDBSW
. . . . . .  . ENDIF
. . . . . .  . CHECK FOR CHANGE IN DATA WORD BLOCK
. . . . . .  . IF EITHER DATA WORD HAS CHANGED OR TRANSMIT DA
. . . . . .  . FLAG IS UP.  (ONLY LOOK FOR TRANSITION OF 0 TO
. . . . . .  . IN BITS 2 THRU 6 IN DATA WORD 1)
. . . . . .  . .THEN
. . . . . .  . .  .LOAD THE INPUT ARRAY
. . . . . .  . .  .PUT THE DATA WORD COUNT ONTO THE RT STATU
. . . . . .  . .  .RESET THE VALUES OF  IOLDDW1 AND IOLDDW2
. . . . . .  . .  .INCREMENT THE WORD COUNTER
. . . . . .  . .  .SET TRANSMIT DATA WORDS FLAG TO ZERO
. . . . . .  . .  .SET THE DATA SENT FLAG
. . . . . .  . .ELSE
. . . . . .  . .  .NO CHANGE IN DATA WORD BLOCK SO PROCEED
. . . . . .  . ENDIF
. . . . . .  . CHECK TO SEE IF INPUT ARRAY IS NON-EMPTY OR RT
. . . . . .  . STATUS HAS CHANGED
. . . . . .  . IF NPPWDS IS GREATER THAN ONE OR RT STATUS WOR
. . . . . .  . HAS CHANGED OR TRANSMIT RT STATUS FLAG IS UP
. . . . . .  . .THEN
. . . . . .  . .  .PACK THE NEW WORDS
. . . . . .  . .  .CALCULATE NPPWDCT THE PP WORD COUNT
. . . . . .  . .  .CALCULATE NBYTE= THE BYTE COUNT
. . . . . .  . .  .ZERO OUT IOACTPP(103)
. . . . . .  . .  .PUT THE PP WORD COUNT ONTO THE FIRST BYTE
. . . . . .  . .  .PUT THE BYTE COUNT ONTO THE SECOND BYTE O
. . . . . .  . .  .PACK THE RT STATUS WORD
. . . . . .  . .  .CALL THE PACKING ROUTINE
. . . . . .  . .  .ZERO OUT THE DATA WORD COUNT
. . . . . .  . .  .ZERO OUT THE T/F BIT IN THE RT STATUS WOR
. . . . . .  . .  .RESET THE VALUE OF IOLDRT
. . . . . .  . .  .RESET THE VALUE OF IOLDBSW
. . . . . .  . .  .SET TRANSMIT RT STATUS WORD FLAG TO ZERO
. . . . . .  . .  .RESET THE DATA AVAILABLE FLAG
```

```
. . . . . .  .SET THE STATUS SENT FLAG
. . . . . .  .ELSE
. . . . . .  .NO CHANGES IN RT WORDS SINCE LAST OLCS CA
. . . . . .  ENDIF
. . . . . ELSE
. . . . . .  OUTPUT BUFFER FULL FLAGS ARE STILL UP
. . . . . .  "OR" THE VALUES OF STATUS SENT AND DATA SENT
. . . . . .  FLAGS ONTO THE APPROPRIATE ERROR WORD BITS
. . . . .  ENDIF
. . . .  ELSE
. . . . .  FLAGS ARE NOT EQUAL SO DON'T PACK DATA
. . . . .  SET BIT IN ERROR WORD
. . .  .ENDIF
. .  .ELSE
. . .  .BIT IS STILL IN PROGRESS SO RETURN
.  . ENDIF
. ELSE
.  . OLCS IN A QUIESCENT STATE SO RETURN
ENDIF
```

```
      SUBROUTINE PACKPP
      ABSTRACT
           PACKPP WRITES OUT THE HEADER WORD IN THE OUTPUT ARRAY
           AND THE NON-ZERO CONTENTS OF THE INPUT ARRAY.
           THIS IS A DUMMY SUBROUTINE FOR THE ACTUAL PACKPP ROUTINE.
      CODING HISTORY
           1. PROGRAMMED  J. MANGES  CSC  DEC 1977
      END OF ABSTRACT
```

```
DO WHILE I IS LESS THAN NPPWDS+240
ENDDO
```

```
      SUBROUTINE SETABIT(JWORD,NBIT,NUM)
      ABSTRACT
           SETABIT SETS A SPECIFIED BIT TO 0 OR 1 IN A GIVEN WORD
           CALLING PARAMETERS- 1. JWORD- WORD IN WHICH BIT IS TO BE SET
                               2. NBIT- BIT NUMBER OF BIT TO BE RESET
                                     (THE FIRST BIT IN WORD IS BIT 0)
                               3. NUM THE RESET VALUE OF THE BIT

      CODING HISTORY
           1.  PROGRAMMED J. MANGES  12/19/77
      END OF ABSTRACT
```

```
      SUBROUTINE READBIT(JWORD,NBIT,NEWWRD)
      ABSTRACT
           READBIT EXTRACTS AND RIGHT JUSTIFIES A GIVEN BIT WITHIN A
           GIVEN WORD
           CALLING PARAMETERS-
                                1. JWORD- WORD CONTAINING BIT TO BE READ
                                2. NBIT- NUMBER OF BIT TO BE READ
                                        (THE FIRST BIT IN WORD IS BIT 0)
                                3. NEWWRD- RIGHT JUSTIFIED RETURN VALUE
                                           OF THE GIVEN BIT
      CODING HISTORY
           1.PROGRAMMED  J. MANGES  CSC  12/19/77
      END OF ABSTRACT
```

SUBROUTINE SPLASH

ABSTRACT

THIS SUBROUTINE CALCULATES SONOBUOY SPLASH POINTS FOR THE OLCS SUBROUTINE. EQUATIONS USED IN THIS ROUTINE ARE TAKEN FROM APPENDIX D OF THE PROGRAM PERFORMANCE SPECIFICATION FOR LAMPS MK III AVIONICS OPERATIONAL PROGRAM. SPLASH ALSO RESETS VALUES IN THE CHUTE ARRAY AND BITS IN THE OLCS DATA WORDS AFTER A SONOBUOY LAUNCH.

TABLE OF VARIABLES

VARIABLE NAME	DESCRIPTION	
HELO(21)	TRUE HELO AIRSPEED IN FEET/SECOND	AMRS
HELO(1)	HELO HEADING IN RADIANS	
HELO(2)	COSINE OF HELO HEADING	AMRS
HELO(3)	SINE OF HELO HEADING	AMRS
VL	SONO LAUNCHER SPEED IN FEET PER SECOND	AMRS
DDC	DEVICE DRAG COEFFICIENT IN FEET SQRD/LB	
HELO(15)	LAUNCH ALTITUDE IN FEET	
C1-C9	EQN CONSTANTS	
WIND(2)	WIND SPEED IN KNOTS	
WIND(1)	WIND DIRECTION IN ANGULAR DEGREES	GRS
HELO(13)	HELO LAUNCH COORDINATES IN FEET	GRS
HELO(14)		

CODING HISTORY

1. PROGRAMMED J. MANGES 11/29/77

END OF ABSTRACT

```
      CHANGE WIND ANGULAR MEASUREMENT TO RADIANS
      CHANGE WIND VELOCITY MEASUREMENT TO FT/SEC
      SET THE VALUE OF DDC  (DEVICE DRAG COEFFICENT)
      RESET BITS IN THE SONOBUOY INVENTORY FIELDS OF THE OLCS
      DATA WORDS
      IF ISELBY IS LESS THAN 10
      . THEN
      .  . SET THE APPROPRIATE BIT IN DATA WORD 1
      . ELSE
      .  . SET THE APPROPRIATE BIT IN DATA WORD 2
      ENDIF
      SET "IN WATER" FLAG FOR THE LAUNCHED BUOY
      RESET LAUNCH INDICATOR BIT IN DATA WORD ONE
      CALL SETABIT(IDATWD1,2,0)
      CALCULATE VS-THE RELEASE TRUE AIRSPEED IN FEET/SEC
      CALCULATE K THE DRAG FACTOR
      .  .  .  .L ARMS SPASH POINT IN FEET
      .  .  .  .T ARMS TIME OF FLIGHT
      CALCULATE THE SONOBUOY SPLASH POINT COORDINATES (GRS)
```

```
      SUBROUTINE UDOASP
      ABSTRACT
           UDOASP UPDATES INPUTS FROM THE ORDNANCE ARM AND SELECT PANEL
           AND RESETS THE APPROPRIATE BITS IN THE OSRU DATA WORDS
           UDOASP ALSO DETECTS SONOBUOY AND TORPEDO MANUAL LAUNCH
           COMMANDS AND SETS FLAGS TO INDICATE TO THE MAIN PROGRAM
           THAT AN ORDNANCE CALCULATION IS NECESSARY.

      CODING HISTORY
           1. PROGRAMMED  J. MANGES CSC  12/20/77

      END OF ABSTRACT
```

```
      INITIALIZE TORPEDO COUNTER
      CHECK FOR SONOBUOY AUTO OR MANUAL SELECT-LAUNCH MODE
      RESET DATA WORD BITS
      CHECK FOR SONOBUOY MANUAL LAUNCH COMMAND
      RESET DATA WORD BIT TWO
      CHECK SONOBUOY SELECT DISCRETES
      CHECK TO SEE IF WANT TO SKIP MANUAL SELECTION THIS CYCLE
      IF SKIP FLAG IS NOT UP
      . THEN
      .   . CHECK THE UNITS CHUTE BITS
      .   . DO WHILE I IS BETWEEN SIX AND FIFTEEN
      .   .   .  .IF IATOTOG(1) BIT I IS ON
      .   .   .  .  THEN
      .   .   .  .  .  SET THE VALUE OF ISELBY
      .   .   .  .  ELSE
      .   .   .  .  .  CONTINUE LOOKING AT UNITS CHUTES BITS
      .   .   .  .ENDIF
      .   . ENDIF
      .   . CHECK THE TENS CHUTE BIT
      .   . IF TENS CHUTE BIT IS ON
      .   .  .THEN
      .   .  .  .INCREMENT THE VALUE OF ISELBY BY TEN
      .   .  .ELSE
      .   .  .  .LEAVE VALUE OF ISELBY BETWEEN 0 AND 9
      .   . ENDIF
      . ELSE
      .   . SKIP MANUAL SELECTION THIS CYCLE AS HAVE RECEIVED AUTO
      .   . SELECTION FROM SUBROUTINE CONTROL
      .   . RESET ISKIP FLAG
      ENDIF
      MAKE SURE TORPEDO LAUNCH BIT IS ZERO
      CHECK MASTER ARM STATUS
      IF MASTER ARM IS ON
      . THEN
      .   . CHECK TORPEDO ARM STATUS
      .   . IF TORPEDO ARM IS ON
      .   .  .THEN
      .   .  .  .CHECK FOR A TORPEDO MANUAL LAUNCH COMMAND
      .   .  .  .(TRANSITION FROM OFF TO ON OF IATOTOG(1), BIT 2)
      .   .  .  .IF IUPS NOT EQUAL TO ZERO
      .   .  .  .  THEN
      .   .  .  .  .  INCREMENT THE TORPEDO COUNTER
      .   .  .  .  .  IF HAVE NOT ALREADY FIRED BOTH TORPEDOES
      .   .  .  .  .  . THEN
      .   .  .  .  .  .  . SET TORPEDO CALCULATIONS NECESSARY FLAG
      .   .  .  .  .  .  . TCALC=1.
      .   .  .  .  .  .  . SET THE TORPEDO LAUNCH BIT IN DATA WORD ONE
      .   .  .  .  .  .  . SET TORPEDO AWAY SIGNAL COUNTER
      .   .  .  .  .  . ELSE
      .   .  .  .  .  .  . HAVE FIRED BOTH TORPEDOES THIS RUN
      .   .  .  .  .  ENDIF
      .   .  .  .  ELSE
      .   .  .  .  .  NO CALCULATION NECESSARY SO PROCEED
      .   .  .  .ENDIF
      .   .  .ELSE
      .   .  .  .TORPEDO ARM NOT ACTIVATED SO NO TORPEDO LAUNCH
```

```
.  . ENDIF
.  . CHECK FOR A SONOBUOY MANUAL LAUNCH COMMAND
.  . IF IANSWER EQUALS ONE
.  .  .THEN
.  .  .  .SET SONOBUOY CALCULATIONS NECESSARY FLAG
.  .  .  .SET SONOBUOY LAUNCH INDICATOR IN DATA WORD 2
.  .  .  .INITIALIZE SONOBUOY AWAY SIGNAL COUNTER
.  .  .ELSE
.  .  .  .NO CALCULATION NECESSARY SO PROCEED
.  . ENDIF
. ELSE
.  . MASTER ARM NOT ACTIVATED SO NO SONOBUOY OR TORPEDO LAUNCH
ENDIF
```

SUBROUTINE TSPLASH

ABSTRACT

THIS SUBROUTINE CALCULATES TORPEDO SPLASH POINTS FOR THE OLCS SUBROUTINE. EQUATIONS USED IN THIS ROUTINE ARE TAKEN FROM APPENDIX D OF THE PROGRAM PERFORMANCE SPECIFICATION FOR LAMPS MK III AVIONICS OPERATIONAL PROGRAM.

TABLE OF VARIABLES

VARIABLE NAME	DESCRIPTION
HELO(21)	TRUE HELO AIRSPEED IN FEET/SECOND
HELO(2)	COSINE OF HELO HEADING
HELO(3)	SINE OF HELO HEADING
VL	TORPEDO LAUNCHER SPEED IN FEET PER SEC
GAMMAL	TORPEDO LAUNCHER ANGLE IN ANGULAR DEGREES
TDDC	DEVICE DRAG COEFFICIENT
HELO(15)	LAUNCH ALTITUDE IN FEET
C1-C9	EQN CONSTANTS
WIND(2)	WIND SPEED IN FEET/SECOND
WIND(1)	WIND DIRECTION IN ANGULAR DEGREES
HELO(13)	HELO LAUNCH COORDINATES IN FEET
HELO(14)	

CODING HISTORY

1. PROGRAMMED J. MANGES CSC DEC. 1977

END OF ABSTRACT

```
CHANGE WIND MEASUREMENT TO RADIANS
CHANGE WIND SPEED MEASUREMENT TO FT/SEC
CALCULATE VS-THE RELEASE TRUE AIRSPEED IN FEET/SEC
CALCULATE K THE DRAG FACTOR
.  .  .  .L ARMS SPLASH POINT IN FEET
.  .  .  .T TIME OF FLIGHT
CALCULATE THE TORPEDO SPLASH POINT COORDINATES
SET TORPEDO SYMBOL ACTIVE FLAG
```

SUBROUTINE WET
ABSTRACT
THIS SUBROUTINE CALCULATES THE WATER ENTRY TIME OF SONOBUOYS FOR THE OLCS SUBROUTINE. EQUATIONS USED IN THIS ROUTINE ARE TAKEN FROM APPENDIX D OF THE PROGRAM PERFORMANCE SPECIFICATION FOR LAMPS III AIONICS OPERATIONAL PROGRAM.

TABLE OF VARIABLES

VARIABLE NAME	DESCRIPTION
DDC	DEVICE DRAG COEFFICIENT
HELO(15)	LAUNCH ALTITUDE IN FEET
TIME	PRESENT TIME IN SECONDS

CODING HISTORY
1. PROGRAMMED J. MANGES CSC DEC, 1977

END OF ABSTRACT

```
      CALCULATE K THE DRAG FACTOR
      . . . .T THE TIME OF FLIGHT
      WATER ENTRY TIME IS EQUAL TO THE PRESENT TIME PLUS THE TIME OF FLI
```

```
      UBROUTINE TWET
      BSTRACT
      .  .  THIS SUBROUTINE CALCULATES THE WATER ENTRY TIME
      .  .  OF TORPEDOS.
      ODING HISTORY
      .  . 1. PROGRAMMED  J. MANGES   CSC  APRIL, 1978
      ND OF ABSTRACT
      CALCULATE K THE DRAG FACTOR
      .  .  .  .T THE TIME OF FLIGHT
      WATER ENTRY TIME IS EQUAL TO PRESENT TIME PLUS THE TIME OF FLIGHT
```

```
SUBROUTINE CONTROL
ABSTRACT
     THIS SUBROUTINE EXTRACTS AUTO SELECT AND AUTO LAUNCH
     INFORMATION FROM INCOMING CONTROL COMMAND DATA WORDS.

CODING HISTORY
     1. PROGRAMMED J. MANGES CSC  12/29/77
END OF ABSTRACT
```

```
EXTRACT AUTO SELECT COMMAND FIELD FROM CONTROL COMMAND DATA WORD
IF IASCFLD IS BETWEEN ONE AND TWENTY FIVE INCLUSIVE
. THEN
.  . RECORD THE CHUTE NUMBER OF THE SELECTED BUOY
.  . SET FLAG TO SKIP ANY MANUAL CHUTE SELECT SETTINGS THIS CYCLE
. ELSE
.  . NO AUTO SELECT COMMAND INDICATED
ENDIF
EXTRACT AUTO LAUNCH COMMAND FIELD FROM CONTROL COMMAND DATA WORD
IF AUTO LAUNCH COMMAND FIELD INDICATES SONOBUOY AUTO LAUNCH
. THEN
.  . CHECK FOR MASTER ARM
.  . IF MASTER ARM IS ON
.  .  .THEN
.  .  .  .SET SONOBUOY CALCULATIONS NECESSARY FLAG
.  .  .  .SET LAUNCH INDICATOR BIT IN DATA WORD 1 TO ONE
.  .  .  .INITIALIZE SONOBUOY AWAY SIGNAL COUNTER
.  .  .ELSE
.  .  .  .NO LAUNCH AS MASTER ARM IS NOT ON
.  . ENDIF
. ELSE
.  . NO CALCULATION NECESSARY SO PROCEED
ENDIF
```

```
      SUBROUTINE UDOH
      ABSTRACT
           THIS ROUTINE GENERATES SONOBUOY AND TORPEDO AWAY SIGNALS
           AND RESETS THE APPROPRIATE BITS WITHIN THE OSRU DATA
           WORDS.
      CODING HISTORY
           1. PROGRAMMED  J. MANCES  CSC  12/22/77
      END OF ABSTRACT
```

```
      SONOBUOY AWAY SIGNAL PROCESSING
      IF COUNTER IS GREATER THAN ZERO
      . THEN
      .  . ZERO OUT SONOBUOY AWAY BIT
      .  . DECREMENT THE COUNTER
      .  . CHECK FOR A ZERO COUNTER
      .  . IF COUNTER IS NOW EQUAL TO ZERO
      .  .  .THEN
      .  .  .  .SET THE SONOBUOY AWAY BIT IN DATA WORD 1
      .  .  .ELSE
      .  .  .  .COUNTER STILL POSITIVE SO CONTINUE
      .  . ENDIF
      . ELSE
      .  . ZERO OUT THE SONOBUOY AWAY BIT IN DATA WORD 1
      ENDIF
      TORPEDO AWAY SIGNAL PROCESSING
      IF COUNTER IS GREATER THAN TO ZERO
      . THEN
      .  . DECREMENT THE COUNTER
      .  . CHECK FOR A ZERO COUNTER
      .  . IF COUNTER IS NOW EQUAL TO ZERO
      .  .  .THEN
      .  .  .  .SET THE TORPEDO AWAY BIT IN DATA WORD 1
      .  .  .ELSE
      .  .  .  .COUNTER STILL POSITIVE SO CONTINUE
      .  . ENDIF
      . ELSE
      .  . ZERO OUT THE TORPEDO AWAY BIT IN DATA WORD ONE
      ENDIF
```

INPUT/OUTPUT EXECUTIVE AND DATA COLLECTION MODULE

(IOEXEC)

```
      SUBROUTINE IOEXEC
      ABSTRACT
           THIS ROUTINE TAKES PACKED DATA FROM THE IIU VIA THE PP
           PROGRAM . HANDS OFF THE PACKED DATA TO THE DATA COLL
           ROUTINE AND THEN UNPACKES THE DATA AND DELIVERS IT
           TO THE VARIOUS REMOTE TERMINAL MODULES.
      PROGRAM HISTORY
           1. PROGRAMMED : ROBERT J. HUBER (CSC)
      END OF ABSTRACT
```

```
DO UNTIL ALL RT INPUT BUFFERS HAVE BEEN SERVICED
.    DECODE THE COMMAND WORD
.    IF RT ADDRESS INVALID
.    . THEN
.    .    SET FLAG TO BYPASS THIS DATA
.    . ELSE
.    .    CONTINUE PROCESSING
.    ENDIF
.    IF RT DATA IS BEING SAVED
.    . THEN
.    .    DO UNTIL ALL DATA STORED IN DATA COLLECTION BUFFER
.    .    .    IF DATA COLLECTION INDEX LESS THAN MAX BUFFER SIZE
.    .    .    . THEN
.    .    .    .    INCREMENT INDEX AND STORE DATA IN D.C. BUFFER
.    .    .    . ELSE
.    .    .    .    SET ERROR FLAG
.    .    .    ENDIF
.    .    ENDDO
.    . ELSE
.    .    DATA NOT COLLECTED FOR THIS RT
.    ENDIF
.    CASE OF SUBADDRESS/MODE (ISAM)
.    .    *ISAM .EQ. 0
.    .    MODE/DISCRETE DATA
.    .    IF CMND IS INIT TERM.INIT PROC. OR INIT SELF-TEST
.    .    . THEN
.    .    .    UNPACK DATA AND INSERT INTO RT INPUT BUFFER
.    .    .    INCREMENT LAST-WORD-IN POINTER (INPTRT)
.    .    . ELSE
.    .    .    MODE/DISCRETE NOT PROCESSED
.    .    .    RESET STATUS SENT FLAG
.    .    END-IF
.    .    *ISAM .EQ. 1
.    .    NORMAL DATA TRANSFER
.    .    IF T/R FLAG IS R-TO-AOP TRANSFER
.    .    . THEN
.    .    .    TRANSFER RT(S OUTPUT BUFFER TO DATACOL BUFFER
.    .    IF RTDATA IS BEING SAVED
.    .    . THEN
.    .    .    DO UNTIL ALL DATA STORED IN DATA COLL BUFFER
.    .    .    .    IF DATA COLLECTION INDEX LESS THAN MAX BUFFER
.    .    .    .    .SIZE
.    .    .    .    . THEN
.    .    .    .    .    INCREMENT INDEX AND STORE DATA IN D.C.
.    .    .    .    .    .BUFFER
.    .    .    .    . ELSE
.    .    .    .    .    SET ERROR FLAG
.    .    .    .    ENDIF
.    .    .    ENDDO
.    .    . ELSE
.    .    .    DATA NOT COLLECTED FOR THIS RT
ENDIF
.    .    . ELSE
.    .    .    TAKE NO DATACOL ACTION
.    .    ENDIF
.    .    DO WHILE DATA IS TO BE UNPACKED
```

```
.    .    .    DO WHILE A WORD IS UNPACKED
.    .    .    END-DO
.    .    END-DO
.    .    *ISAM .EQ.2
.    .    INCREMENT *IP* FOR A STRAGGLING STATUS WORD
.    .    *ISAM .EQ. 3
.    .    RETRANSMIT LAST MESSAGE
.    .    *ISAM .EQ. 4
.    .    TRANSMIT LAST COMMAND
.    .    THESE COMMANDS NOT PROCESSED BY THE
.    .    RT MODULES
.    .    RESET OUTPUT BUFFER FULL (STATUS ONLY) FLAG
.    .    *ISAM .EQ. 5
.    .    CONTROL COMMAND DATA TRANSFER
.    .    UNPACK AND INSERT COMMAND WORD INTO RT-S BUFFER
.    .    INCREMENT LAST-WORD-IN POINTER FOR RT
.    .    UNPACK/INSERT CONTROL COMMAND DATA WORD INTO RT-S BUFFER
.    .    INCREMENT THE AYK OUTPUT BUFFER POINTER
.    .    AROUND STRAGGLING STATUS WORD
.    .    *ISAM .EQ. 6
.    .    MULTI-MESSAGE TRANSFER
.    .    DO WHILE DATA IS TO BE UNPACKED
.    .    .    DO WHILE A WORD IS UNPACKED
.    .    .    END-DO
.    .    END-DO
.    ENDCASE
.    INCREMENT THE AYK OUTBUFFER(IAYKBUF) POINTER
END DO WHILE
```